AD-A068 958 NAVAL UNDERSEA WARFARE CENTER SAN DIEGO CA F/G 17/1

A COMPUTER PROGRAM TO SIMULATE DATA FROM THE ACTIVE SONAR PROCE--ETC(U)

NOV 68 W J WILSTERMAN

UNCLASSIFIED NUWC-TN-211 NL

| OF |

AD
A068958

END
DATE
FILMED
7-79
DDC

Ships OOVI
Carey Smith

NUWC-TN-211

14

MOST Project -3

C3 Keb

B file

002043

NAVAL UNDERSEA WARFARE CENTER

11 NOV[redacted]68

1

12 88 p.

LEVEL II

6

A COMPUTER PROGRAM TO SIMULATE DATA FROM THE ACTIVE SONAR PROCESSOR

10 by W. J. Wilsterman, Jr

San Diego, California

17 SUBPROJECT NO. SF11-121-100

TASK NO. 11197

16 F11121

62711 N

002043

403 023

PREFACE

This memorandum describes a computer program which was developed by the author to simulate data from the active sonar processor for the purpose of developing a computer program for the automatic sonar processor. This program was written for Code 603, Naval Undersea Warfare Center, San Diego Division, Subproject SF 11 121 100, Task 11197, (NUWC Problem E119). This memorandum has been prepared because it is believed that the information may be useful in this form to others at the Naval Undersea Warfare Center (NUWC) and to a few persons outside NUWC. This memorandum should not be construed as a report as its only function is to present for the information of others an explanation of the above mentioned computer program.

ABSTRACT

The computer program to simulate data from the automatic sonar processor (ASP), known as SYSTEM SIMULATASP, generates both noise and target data for subsequent analysis and processing by the ASP computer program. The program, SYSTEM SIMULATASP, was written by the author for Code 603 in conjunction with the development of the ASP computer program. The data formatted by the SYSTEM SIMULATASP is identical to the format used by the ASP program. Noise and target levels, target position and number, number of pings, and mode selection are all arbitrary parameters of SIMULATASP. SIMULATASP is capable of generating data to test every operating mode of the ASP.

TABLE OF CONTENTS

I. INTRODUCTION

The purpose of this computer program, SYSTEM SIMULATASP, is to simulate input data for testing the Active Sonar Processor (ASP) computer program. For further information regarding the ASP see reference 1 in the back. Input to SIMULATASP is by means of data cards. Since a portion of the ASP computer program (ASPP) must input data from magnetic tape, the output data generated by SIMULATASP is placed on magnetic tape.

The primary function of the ASPP is to process active sonar data received from the ASP hardware. From the processing of this data, clusters (targets and false alarms) will be identified, tracked and displayed. The information received from the ASP consists of three items: 1) Signal level data, 2) Operating mode, and 3) Correlation data. The formats for operating mode and correlation data are described later in this document. (SIMULATASP is not concerned with signal level data). The ASPP is modularized into three major subroutines: 1) The EXECUTIVE subroutine, including I/O and processing, 2) The TRACK subroutine, and 3) The DISPLAY subroutine.

SIMULATASP is designed to simulate the correlation data from the ASP. Other data besides correlation data is formatted by this program such as operating mode, but the main function of the program is to simulate this correlation data. This divides itself into two problems since correlation data is of two kinds: noise data and target data. This leads directly into the theoretical description of the data. Part II of this report covers the theory of noise formation and the theory of target formation.

II. THEORETICAL DESCRIPTION

A. Theory of Noise Formation

The theory of noise formation is based upon one assumption. It has been found empirically that the distribution of sonar noise data at the output of an envelope detector in a clipper correlator processor follows an amplitude distribution approximately equal to a Rayleigh distribution. This was borne out by studies done by R. F. Ayala in his examination of zero threshold sonar data and by L. Boelter in his work involving the measurement of processing gain of a correlation system. Messrs. Ayala and Boelter, both of NUWC Code 603, noted empirically that the noise at the output of the correlator's envelope detector followed a Rayleigh distribution. The Rayleigh distribution is given by the following formula:

$$p(r) = \frac{r}{N} \exp(-r^2/2N) \qquad (1)$$

where $p(r)$ is the probability density function for the envelope r of noise at the output of the amplitude correlator. N is the mean-squared amplitude at the input of the envelope detector.

The probability of finding a noise data sample between r_1 and r_2 is consequently given by the following integral:

$$\int_{r_1}^{r_2} p(r)\, dx = \frac{1}{N} \int_{r_1}^{r_2} r \exp(-r^2/2N) dr \qquad (2)$$

The probability density function is normalized as can be shown by integrating from 0 to ∞:

$$\int_0^\infty p(r) dr = \int_0^\infty \frac{r}{N} \exp(-4^2/2N) dr \qquad (3)$$

$$= -\exp(-r^2/2N)\Big|_0^\infty = -0+1 = 1 \qquad (4)$$

The probability, P(x), is given by:

$$P(x) = \int_0^x p(r)dr \tag{5}$$

P(x) is the probability that $r \leq x$ and is called the cumulative probability function. Let us solve for this function, P(x). Substituting (1) into (5) yields:

$$P(x) = \int_0^x \frac{r}{N} \exp(-r^2/2N)dr \tag{6}$$

This can be integrated directly as shown in equation (4):

$$P(x) = -\exp(-r^2/2N)\Big|_0^x \tag{7}$$

The result is given by the following equation:

$$P(x) = 1 - \exp(-x^2/2N) \tag{8}$$

We can find the most probable value for the probability density function by taking the first derivative of equation (1) and setting it equal to zero.

$$p(r) = \frac{r}{N} \exp(-r^2/2N) \tag{9}$$

$$\frac{dp(r)}{dr} = \frac{1}{N} \exp(-r^2/2N) + \frac{r}{N} \left(\frac{-2r}{2N}\right) \exp(-r^2/2N) = 0 \tag{10}$$

$$\frac{1}{N} \exp(-r^2/2N) - \frac{r^2}{N^2} \exp(-r^2/2N) = 0 \tag{11}$$

$$\frac{1}{N} - \frac{r^2}{N^2} = 0$$

$$\frac{1}{N} = \frac{r^2}{N^2}$$

$$N = r^2 \tag{12}$$

If $r = R_{max}$ is the most probable value of $p(r)$, then by equation (12), R_{max} is given by:

$$R_{max} = \sqrt{N} \tag{13}$$

As x ranges from 0 through ∞, $P(x)$ ranges from 0 to 1. Furthermore, by equation (8), $P(x)$ is a strictly monotonically increasing function. If a random number, X, is selected from 0 to 1, there is a unique number, λ, such that $X = P(\lambda)$. Moreover, if a large number of such random numbers is selected, and related to their corresponding λ values, these λ values will have a distribution density function equal to $p(x)$, i.e., these values will constitute a Rayleigh amplitude distribution.

Note that the cumulative probability function $P(x)$ given by Equation (8) is determined by only one arbitrary constant, N. Equation (13) shows that N is related to the most probable value, R_{max}. Thus the position of R_{max} can be shifted simply by changing the value of N.

Noise amplitude data is generated in the following manner: R_{max} is determined by a card input parameter, LAMBDA. A table is generated for $P(x)$ using Equation (8) and letting x vary from 1 to 4095, the maximum amplitude value. A random number generator selects random numbers from 0 to 1 and they are compared to items in the table. When these numbers are found to lie between two values of the table, the index number of the upper interval is selected as the corresponding noise amplitude value.

B. Theory of Target Formation

Targets could perhaps be simulated by a number of different models. For the purpose of programming, just one of these models was selected, and just this model will be discussed herein.

In order to specify a target in SIMULATASP, the following parameters must be given: 1. the data sample rate, 2. the target range, 3. the target bearing, 4. the target speed, 5. the target course, 6. the target bearing standard deviation, 7. the target doppler standard deviation, and 8. the target amplitude level.

The data sample rate depends upon the ASP operating mode selected. The various possible sample rates are 100, 200, 400, 800, 1600 and 3200 samples/second. The number of sample pulses used in generating the target is dependent upon the sample rate. The target model is assumed to produce a cluster 10 milliseconds long. Thus, if the sample rate is 100 samples/second, the target spans one sample pulse, and if the sample rate is 3200 samples/second, the target spans 32 sample pulses.

The position of the target is given by the target range and by the target bearing information. The range in the computer is in range counts and the bearing is in degrees to the nearest minute. The target is assumed to move from ping to ping so that during each ping this range and bearing information must be updated.

Updating the target position is dependent upon the target course, the target speed, and the time between pings. A somewhat modified cartesian scale is used for the updating equations. Consider the x-axis to point east and the y-axis to point north. Let the reference submarine heading be north (which it is always defined to be throughout the formation of this data). Let the target bearing be given as θ, which is the angle the range vector makes with the y-axis, where θ is taken in a positive sense in a clockwise direction from the y-axis. Let R be the range in yards.

Then the x and y coordinates are given by

$$x = R \sin \theta$$

$$y = R \cos \theta \tag{14}$$

Let φ be the target heading, where φ is taken in a positive sense in a clockwise direction from the y-axis. Let v be the target speed in knots. Let t be the time in seconds between pings. Let Δx and Δy be the change in the x and y coordinates of the target between pings. Λx and Δy are then given by

$$\Delta x = vt \sin \varphi$$

$$\Delta y = vt \cos \varphi \tag{15}$$

The new position of the target after one ping would be:

$$x' = x + \Delta x$$

$$y' = y + \Delta y \tag{16}$$

A new range R' and bearing θ' must be computed from these new x and y coordinates. These are given by

$$R' = (x'^2 + y'^2)^{\frac{1}{2}} \tag{17}$$

$$\theta' = \arctan \frac{x'}{y'} \tag{18}$$

The doppler, d, of a target in knots is given by

$$\mu_d = -v \cos (\varphi - \theta) \tag{19}$$

Note the minus sign in the right-hand member of the equation. This is due to the definition that a closing range shall be a positive doppler and an opening range shall be a negative doppler.

Each mode of the ASP has a different set of parameters which specify the mode. The pertinent parameters for simulation purposes are: S, the sample rate; b, the number of beams; f, the number of doppler filter teeth per channel; r, the number of reference channels (usually only 1 or 2); and D, the total doppler coverage in knots. For each sample,

target amplitudes must be generated for each beam and for each doppler, i.e., b·f·r amplitudes in all. In each ping a target has a specific doppler, a specific range, and a specific bearing.

Placing the target at the correct range is the simplest target parameter to simulate. The noise data is generated until the target range is reached, then the target information is generated. When the target range has been surpassed, generation of noise data is continued.

Simulation of doppler and beam information is not so simple, however. It is desired to have a model which assigns highest target amplitudes to doppler and beam values closest to the doppler and beam of the simulated target, intermediate amplitudes to doppler and beam values close by the target doppler and beam, and low amplitudes to doppler and beam values far from the target values. Therefore assume that each doppler and each beam has an output response identical to the Gaussian probability density function. Thus if A_{TL} is the target amplitude level and A_T is the target amplitude, then A_T is given by:

$$A_T = A_{TL} \exp(-\tfrac{1}{2}\alpha^2) \exp(-\tfrac{1}{2}\beta^2) \tag{20}$$

$$A_T = A_{TL} \exp(-\tfrac{1}{2}(\alpha^2 + \beta^2)) \tag{21}$$

where α = doppler deviation from the target doppler and β = beam deviation from the target bearing. The final output amplitude A, on a particular doppler, beam and range has a noise component and a target component. A_T means the component of the final output amplitude due to the target. A_{TL} is the target level of a target and is given by an input parameter. A_{TL} is attenuated by the factor shown in equation (21) to become A_T. The target component, A_T, has lower values as α and β

are increased in magnitude and reaches its maximum value, A_{TL}, when $\alpha = \beta = 0$. After A_T has been computed for a given target word, a random noise amplitude word is added to it so that the final output amplitude is given by

$$A = A_T + A_N \tag{22}$$

where A_N is the random noise amplitude.

α in equation (21) is given by:

$$\alpha = (d + \mu_d \frac{2f}{D} - \frac{f-1}{2}) \frac{1}{\sigma_D} \tag{23}$$

where d = current doppler of the word being formatted

μ_d = mean doppler of the target

f = number of dopplers

D = doppler extent in knots

σ_D = doppler standard deviation

Changing the value of σ_D allows the operator to control the attenuation of target amplitude over doppler filters. α is in units of standard deviations from the mean.

β in equation (21) is given by:

$$\beta = (\beta - \mu_\beta/2 - \frac{b-1}{2}) \frac{1}{\sigma_\beta} \tag{24}$$

where β = current beam of the word being formatted

$\mu\beta$ = mean beam of the target

b = number of beams

σ_β = beam standard deviation

β like α is also in units of standard deviation from the mean.

III. SYSTEM INPUT

Flow charts exist in the Appendix in the back for all subroutines discussed in this document. For further information or clarification of these subroutines, please refer to these diagrams.

A. Card Input

Input to SIMULATASP is 80 column IBM cards only. The input is translated by the CARDINPUT S/R. Only four types of cards are recognized; 1) mode cards, 2) target cards, 3) end cards, and 4) go cards. Card types are specified by columns 11-15.

1. Mode Parameters

A mode card is specified by the word MODE in Field 2 (columns 11-15). The mode number is punched in Field 3 (columns 16-20). There are 96 modes altogether which are recognized by the mode recognition table. The modes range from A1 to E18 and are described in Reference 1. There are nine other parameters on the mode card. They range from Field 4 to Field 12; they are all left justified positive decimal integers. The following parameters are listed on the mode card: LAMBDA, PINGS, PTIME, TRTIME, LTHRESH, DOPBEAMOP, RTIME, PINGNUM and FILE. LAMBDA is punched in Field 4, PINGS in 5, and so on.

LAMBDA contains the value of the maximum amplitude probability. It is the amplitude correlation value at which p(x) of Equation 12 reaches a maximum. PINGS contains the number of pings for which one wishes to format data. Each ping of data is one file of data on magnetic tape. PTIME is the time between pings in seconds. TRTIME is the transmit receive delay time in seconds. This parameter simulates the time the

computer must wait before the ASP hardware begins to send it data after it has transmitted a ping into the medium. RTIME is the receive time in seconds. This simulates the time the receiving gates are left open and during which the computer receives data. Although there is no specific check or error printout, PTIME should be greater than the sum of TRTIME and RTIME.

LTHRESH is the L-threshold of the ASP. It represents the value below which noise data is not sent to the computer. In SIMULATASP, amplitude correlation data below LTHRESH are not formatted nor stored in the CDATA data table. PINGNUM contains the ping number of the first ping to be formatted by the program. The second ping will have the value PINGNUM+1 and so on. FILE contains the file number of the output upon which one wishes to start formatting data. SIMULATASP requires two output tapes, the final output tape being on tape unit M4, the scratch tape on unit M3.

Parameter DOPBEAMOP is the doppler-beam option. It has four possible values. When zero (usually it is zero), it represents the all dopplers, all beams mode. One represents the best doppler, all beams mode. Two represents all dopplers, best beam mode, and three represents best doppler, best beam mode. No other values are allowed.

The selection of the mode number results in the translation of the following parameters: SRATE, BEAMS, DOPPS, DSPREAD, and FF. SRATE is the sample in hundreds of samples/second and may be 1, 2, 4, 8, 16 or 32. BEAMS is the number of beams in that mode. DOPPS is the number of doppler reference channels. DSPREAD is the doppler spread in knots. FF is the number of doppler filter teeth per reference channel. It may differ from DOPPS when two reference channels have been selected.

2. <u>Target Parameters</u>

A target card is specified by the word TARGET in Fields 2 and 3. There are seven target parameters altogether, which are punched from field 4 to Field 10. The parameters in order are range, bearing, speed, course, doppler standard deviation, bearing standard deviation,and target amplitude level. Range is given in 10 yard increments. Higher resolution than 10 yard increments is not necessary for a target input parameter. This was done to keep the range parameter within five digits, the size of the parametric field on the data card. The range, when translated from the card, is stored in range counts, which give roughly quarter-yard precision to the range. Thus, to indicate a range of 20,000 yards, one would punch 2000 in Field 4. Bearing is given in compass degrees, from 0 to 359. Dead ahead is due north. Speed is given in tenths of knots; to indicate a speed of 15 knots, one would punch 150. Target amplitude level is an integer value ranging from 0 to 4095 full scale.

3. <u>Other Cards</u>

Two other types of cards are recognized by the input card translator, go cards and end cards. A go card is signified by punching GO in Field 2. No other information need be on the card. A go card causes the final processing of input parameters and a return from CARDINPUT to begin the formatting of ASP simulation data. An end card is signified by punching END in Field 2. It causes the final processing of the ASP data. Notably, an end of file is written on M3, the tape is rewound and the SETMTHRESH subroutine is executed. After this, the run is finished. Effectively, an end card terminates a run.

B. CARDINPUT S/R

The function of CARDINPUT is to read a set of parameter cards and to translate them thus initiating a data formatting run.

A typical run deck for SIMULATASP is two library cards for FLOP and PRINT followed by a load card for the program, followed by transfer to SIMULATASP, followed by the parameter cards. The first card of the deck must be a job card. FLOP and PRINT are currently loaded at 600 and 2520 respectively while SIMULATASP is based at 7450. The parameter cards typically start with a mode card followed by one or more target cards. There cannot be more than ten target cards. The target cards are followed by a go card. If it is desired to format more than one mode on a tape, this sequence may be repeated to the limit of that which can be stored on tape. An end card after the last go card terminates the formatting process. A second end card tells the MONITOR that the job is terminated.

Immediately following the transfer to CARDINPUT from the executive, it transfers to the CARD S/R and reads one card into CARDBF, a 16-word table used only for this purpose. Field 2 of the card is entered in Q and Q is compared with all the words of the IDFIELD table. IDFIELD is a four word table containing the recognition codes for the four distinct types of input cards. If Q is not equal to any of these codes, the following is typed on the flexowriter; ILLEGAL IDFIELD. Otherwise control is transferred to the specified section.

If a mode card has been read, Field 3 of CARDBF is entered in Q and a search of the mode table is initiated. MODES is a 96 word table which contains the recognition codes for all 96 modes. If a code identical to Q is not found in MODES then the following is typed on the flexowriter; ILLEGAL MODE NUMBER. The program then stops at the beginning of CARDINPUT. If an identical code is found then the corresponding element

in the AA1 table is entered in Q. AA1 is a 96 word table corresponding to MODES which contains all the mode information relating to that particular mode. The mode index, B7, is stored in MODENUM. The data in Q is unpacked and stored in DSPREAD, FF, BEAMS, SRATE and DOPPS. The mode data is packed in the following format:

29 — 22	21 — 15	14 — 12	11 — 6	5 — 0
DSPREAD	FF	RR	BEAMS	SRATE
8 Bits	7 Bits	3 Bits	6 Bits	6 Bits

RR is the number of reference channels. This information, RR, is not stored. DOPPS is stored as the product of FF and RR. PSHIFT is the value by which the sample pulse is incremented each sample pulse and is computed by PSHIFT = 32/SRATE. 8 is then stored into PP and control is transferred to TRANPS to translate the other nine parameters on the mode card.

After the mode card parameters have been translated, they are transferred from the PARAM table to the parameter variables starting at LAMBDA. Negative values in PARAM are not transferred to the LAMBDA area.

BBB is the constant 2N in equation (8) for P(x). It is computed by

$$BBB = 2 \cdot (LAMBDA)^2$$

After the computation of BBB, control is transferred to the MODETEST S/R to print out the mode parameters. TARGX, which contains the number of targets in the TARGETS table is cleared and control is transferred to the beginning of the subroutine.

If a target card has been read, 6 is stored in PP and control is transferred to TRANPS, a subroutine which translates the parameters. After the parameters have been translated, TARGX is entered in B7, which

now acts as the target parameter storage index. The parameters are tested individually and all positive target parameters are moved from their location in the PARAM table to their proper location in the TARGETS table.

TARGETS is a horizontal 10 item, 4 words per item table. It has the following format:

Word 0:	TARGET RANGE	(whole word)
Word 1:	TARGET BEARING	SPEED
Word 2:	COURSE	AMPLITUDE LEVEL
Word 3:	DOPPLER SIGMA	BEARING SIGMA

The target range is converted from 10-yard increments in PARAM to range counts. Target bearing and target course are converted from compass degree values to values ranging from +180° to -180°. Furthermore, target bearing is scaled up 6 bits before storage in TARGETS. Target bearing information must be scaled because this data is updated from ping to ping, and greater precision than one degree is required for this information to be accurate. Target speed is stored in tenths of knots and doppler and bearing sigma are stored in tenths of filter values. When these values have been stored into the TARGETS table, control is transferred to the beginning of CARDINPUT and another card is read.

If a go card has been read, TARGCX is computed if TARGX has a non-zero value according to the following formula:

TARGCX = TARGX-1

TARGCX is the target control index ; it is used to control indexing through the TARGETS table.

PINGNUM is stored in PINGNO. PINGNO is the first word of a four word record which begins every ping of data. This record consists of PINGNO, TRANINFO, TRANTIME, and DATE. SPSTART and SPFINISH are the sample pulse starting time and the sample pulse finishing time in range counts. They are computed from TRTIME and RTIME by the following formulae:

$$SPSTART = 3200 \cdot TRTIME$$

$$SPFINISH = SPSTART + 3200 \cdot RTIME$$

Tape unit M3 is rewound to insure its being at load point.

The parameters for TRANINFO are formatted and packed into TRANINFO. TRANINFO standing for transmitted information contains crucial information regarding the operating functions of the ASP. It contains such information as the mode number, the major and minor modes, the doppler-beam option modes, the operating range of the hardware and the current ping number (only the lower order bits). TRANINFO format is as follows:

	29 27	26 23	22 18	17-16	15-14	13-12	11	10 0
TRANINFO	RANGE SCALE	MAJOR and MINOR MODES	MODE NUMBER	NOT USED	DOP BEAM OPTION	N-T N-P	N U	PING NUMBER
	3 Bits	4 Bits	5 Bits	2 Bits	2	2		11 Bits

Range scale is given in the three bit field from bit 27-29. Range scale may be to 5, 10, 25, 50, 75 or over 75 miles. Major and minor modes are given in the four bit field from bit 23-26. The mode number which is computed by MODENUM+1 is in the five bit field from bits 18-22. Bits 11, 16 and 17 are not used. The doppler beam option is in bits 14 and 15. Bits 12 and 13 contain normalized threshold, normal ping data; both are set. The ping number is given in the bits from 0-10 from the 11 lowest order bits from PINGNO.

The computer clock time is stored in TRANTIME and an arbitrary date is set in DATE. Control is transferred to TARGTEST to print out the target parameters, and afterwards, control is transferred back to the executive. If an end card has been read, control is transferred to the executive at the label SIMUL2.

C. Test and Utility Routines Subordinate to CARDINPUT

CARDINPUT uses three subroutines in the process of performing its function: MODETEST, TARGTEST and TRANPS. MODETEST prints the mode parameters on the printer; TARGTEST prints the target parameters; TRANPS tmanslates the card buffer information from XS-3 codes to its octal equivalent.

1. MODETEST

The subroutine first clears PRTBUFF, a 24-word buffer used in conjunction with MONPRINT. It then moves Fields 2 and 3 from CARDBF to PRTBUFF, transfers control to MONPRINT and prints out the mode number. It prints the values of DSPREAD, FF and BEAMS on the 2nd line. The value printed out for DSPREAD is twice the actual value. SRATE, DOPPS and LAMBDA are printed on the 3rd line; PINGS, PTIME and TRTIME are printed on the 4th line; LTHRESH, DOPBEAMOP and RTIME are printed on the 5th line; PINGNUM, FILE and BBB are printed on the 6th line. MODETEST returns to the calling routine.

2. TARGTEST

The TARGTEST S/R prints out the target parameters of each target. Immediately upon entry, it generates a top of form to the printer. The parameters SPSTART, SPFINISH and PINGNO are printed in the first line. The second line is the target parameter test loader. If TARGX is zero, indicating no targets, TARGTEST returns to the calling

routine. Otherwise it converts the target speed and bearing of each target as indexed by B6 from fixed point to floating and prints one line containing every target parameter. TARGTEST returns to the calling routine after each parameter of each target is printed. The parameters printed of each target are the index number, the range, the bearing, the speed, the course, the amplitude level, the doppler sigma, and the range sigma.

3. TRANPS

This procedure translates XS-3 coded, positive decimal numbers to octal and packs it in table PARAM. XS-3 codes must be used because the CARD S/R reads all characters from a card in XS-3 codes. Input to TRANPS is the cell PP. PP is an index control parameter which controls the number of parameters to be translated by TRANPS. B1 is the parameter index; B2 is the pack index; and B3 is the digit counter. PARAM is the output table.

Initially PARAM, a ten word table, B1 and B2 are cleared. The input word to be translated is taken from the CARDBF area starting at Field 4 when B1 = 0, being indexed by B1. Field 5 would be taken next and so on. The input word is entered in Q. It is unpacked character by character and stored in six bit codes in the five words of table PACK. The first character of PACK is tested to see whether it is an asterisk. An asterisk in an input parameter field indicates that the parameter in core corresponding to it is not to be modified. To indicate this in PARAM, a negative zero is stored in the B1st item of PARAM. Then the subroutine jumps to the point at which B1 is interrogated for equality to PP.

If the first character is not an asterisk then all the characters in PARAM are tested to see whether they are XS-3 coded numbers. If any of then are not numbers then the flexowriter types ILLEGAL PARAMETER CARD and the computer stops. B2 keeps track of the number of

characters until a blank is reached. When a blank has been reached, the characters are translated to binary coded decimal numbers and are packed together as the octal equivalent of a decimal number. The number so formed is stored in the B1st item of PARAM.

PP is then interrogated. If B1 equals PP then TRANPS returns to the calling routine. Otherwise B1 is incremented by 1 and the next field is pulled from CARDBF. PP causes the translation of one more parameter than its value. If PP is 5, TRANPS will translate 6 parameters.

IV. SYSTEM CALCULATION

The majority of the computations and computer usage time is spent in the FORMNOISE and FORMDATA subroutines. FORMNOISE calculates the cumulative probability table for noise amplitude distribution as given by equation (11). FORMDATA formats and thresholds the noise and target data and writes the data on tape.

A. FORMNOISE

This subroutine computes the cumulative probability table for noise amplitude distribution. The input is BBB. The output is the NOIPROB table. NOIPROB is a 10000 octal item table, one word per item, each item being scaled up 28 bits. Essentially, this subroutine computes values for P(x) in equation (8), ranging x between 0 and 4095 (decimal).

The actual computation is done in floating point because it was found that fixed point computation could not be made precise enough; it is straightforward. Upon entry, NOIPROB is cleared, B1 is set to 1. Two numerical constants are converted from fixed point to floating: 1, and -1, and are stored in CON1 and CONMS1 respectively. BBB is converted from fixed point to floating and is stored in BBBFL. B1 is indexed from 1 to 4095 and is used as the x-value. It is converted from fixed to floating and stored in LAMBDAFL. The first term of (8), TERM1 is computed by:

$$TERM1 = (LAMBDAFL)^2$$

The second term, TERM2, is computed by:

$$TERM2 = (TERM1)(BBBFL)$$

The third term is computed by:

$$TERM3 = (TERM2)(CONMS1)$$

TERM3 is the argument of the exponent:

$$TERM3 = \exp(TERM3)$$

P(x) is computed by:

Z2 = CON1-TERM3

Z2 is converted from floating point to fixed and stored in the B1st item of NOIPROB. B1 is incremented, another value is computed until B1 = 4095. The subroutine then transfers control to the calling routine.

B. FORMDATA

This procedure creates noise and target data and writes it on tape. B1 is used as the data output index. B2 is the ping counter index. B3 is the matrix array index.

Housekeeping is done upon entry. MATRIX contains the number of elements in the doppler-beam array; it is the number of noise data which must be generated for each sample pulse. MATRIX is given by:

MATRIX = (DOPPS)(BEAMS)

MCTR is the matrix array control index; its value is one less than MATRIX and is given by:

MCTR = MATRIX-1

DOPDIF, standing for doppler difference constant, is a constant used in centering the formatted doppler between 37 and 40 (octal). Its value is given by

DOPDIF = (64-FF)/2

(64 decimal).

BEAMDIF is the beam difference constant. It is used to center the formatted beam between 7 and 10 (octal). Its value is given by:

BEAMDIF = (16-BEAMS)/2

FFX is the number of doppler filter teeth/reference channel control index; its value is given as 1 less than that of FF. Two doppler constants are used to values in the 2nd and 3rd terms of equation (23). DOPCON1 and DOPCON2 are given by

DOPCON1 = 2(FF)/DSPREAD

DOPCON2 = (FF-1)/2

After the housekeeping, the ping looping cycle is started off by writing the initial four word record on tape. The value of SPSTART is sent to SAMPULSE.

The ARRAY table is a variable length one word per item table having a maximum length of 2000 octal words. Its length is specified by the value of MATRIX. It contains the amplitude values for all dopplers and beams of each sample pulse. The ARRAY table is cleared. B3 is cleared.

A 22-bit random number is generated by calling upon RANDOM and is placed in A. The NOIPROB table is searched item by item from the beginning for a value greater than or equal to A, indexing B7. When this value is found, the B7 index is stored in the B3rd item of ARRAY. This process functions as the random generator of a noise datum whose probability distribution is specified by a Wien distribution. B3 is compared to MCTR. If B3 is not equal to MCTR, B3 is incremented and control is transferred to RANDOM, to generate another 28 bit random number and to continue the process. If B3 is equal to MCTR, control is transferred to the target generation mode.

In the target generation mode TARGX is tested. If TARGX is zero, this mode is skipped; control is then transferred to the doppler-beam option test mode. If not, B4 is cleared. A and Q are entered with the

sample pulse target test limits. The target range of each target, as indexed by B4, is checked. B4 is indexed through TARGCX. If no target is found within the test limits, control is transferred to the doppler-beam option test mode. Otherwise the target generation mode continues.

The target doppler is computed according to equation 19. The target bearing is subtracted from the target course; the difference is converted from degrees to radians and stored in SAVEQ. SAVEQ is converted from fixed to floating by FLOP. FLOP computes the cosine and converts it back to fixed point. SAVEQ is multiplied by the target speed; the result is stored in MEANDOP.

The next phase of the target generation mode is repeated as many times as there are items in ARRAY. Beam and doppler are specified by dividing B3 by DOPPS, by storing the quotient in CBEAM, and by storing the remainder in CDOPPLER.

The beam deviation argument is computed next. Q is computed as follows:

$$Q = \mathrm{BEAMS}-1+(2\cdot\mathrm{CBEAM})+\mathrm{TARGETS}(\mathrm{BEARING}+\mathrm{B4})$$

The value of Q is then checked. If Q is greater than 180, then Q is decremented by 360. The result is that Q contains a number lying between -179 and +180. The bearing deviation argument is computed from Q by

$$\mathrm{ARG} = (Q\cdot 5/\sigma_B)^2$$

where σ_B is the target bearing standard deviation.

The doppler deviation is computed from MEANDOP and CDOPPLER and entered in Q. Q is computed as follows:

$$Q = (\mathrm{DOPCON1})(\mathrm{MEANDOP})+\mathrm{CDOPPLER}-\mathrm{DOPCON2}$$

The doppler argument, Z1, is computed from and is given by:

$$Z1 = (10 \cdot Q/\sigma_D)^2$$

where σ_D is the target doppler standard deviation. The final exponential argument is given by:

$$Z2 = (ARG+Z1)/2$$

FLOP is used to compute the exponential value of Z2 and is stored in SAVEQ:

$$SAVEQ = \exp(Z2)$$

The product of the target amplitude and SAVEQ is entered in Q. The value of Q at this point is equivalent to the A_T of equation (20). Finally the B3rd element of ARRAY is incremented by Q. The result in ARRAY element is equivalent to the A of equation (22). Control is transferred to the TARGAMPT subroutine which prints out the value of A_T.

B3 is then checked. If B3 does not equal MCTR then B3 is incremented and control is transferred to the beam and doppler specification phase. Otherwise B4 is checked. If B4 is not equal to TARGCX then B4 is incremented and control is transferred back to the beginning of the target generation mode at the point where the target range of each target is checked. Otherwise control is transferred to the doppler-beam option mode.

The doppler-beam option mode consists essentially of a four-way switch. The branch of the switch is determined by DOPBEAMOP, whose value may range from 0 to 3. DOPBEAMOP = 0 represents the selection of all dopplers and all beams. B3 is indexed from 0 to MCTR and PACKITEM is called upon MATRIX times. Afterwards, control is transferred to the end of sample pulse mode.

DOPBEAMOP = 1 represents the selection of the best doppler on each beam. This mode must have two indices, one for beams, the other for dopplers. CBEAM is the beam index. CBEAM is initially cleared. Each doppler on a beam is examined. When the maximum amplitude is found then

control is transferred to PACKITEM to store the information. CBEAM is incremented; the process is repeated with the best doppler on each beam sent to PACKITEM until all beams have been sent. Control is then transferred to the end of sample pulse mode.

DOPBEAMOP = 2 represents the selection of the best beam on each doppler. This is similar to the previous case except that dopplers and beams are exchanged. When the last doppler is sent to PACKITEM, control is transferred to the end of the sample pulse mode.

DOPBEAMOP = 3 represents the selection of the best doppler on the best beam. In this case, all the elements of ARRAY are checked and only maximum amplitude element is sent to PACKITEM to be processed. Control is then transferred to the end of the sample pulse mode.

At the end of the sample pulse, SAMPULSE is incremented by PSHIFT. SAMPULSE is compared to SPFINISH; if SAMPULSE is less than SPFINISH then control is transferred to the beginning of the sample pulse mode where the ARRAY table is cleared. Otherwise control is transferred to the end of ping mode, the end of the ping having been reached.

At the end of a ping, PINGNO and TRANINFO are incremented. If B1 is not equal to zero, indicating unstored output data in the CDATA table, the CDATA table is written on tape. Otherwise and next, an end of file is written on tape. The CDATA table is cleared and B1 is cleared.

TARGX is checked. If TARGX is zero, indicating no targets in the TARGETS table, control is transferred to the end of ping test mode. Otherwise control proceeds to execute the target update mode.

B4 is cleared first in the target update mode. Each target must be moved from ping to ping. B4 is the target control index. The target bearing is extracted and is converted from degrees to radians.

The sine and cosine of the bearing is computed and stored in SINPH1 and COSPH1 respectively. The X-coordinate of the target is computed by

EXDIST = (SINPH1)(RANGE)

The y-coordinate of the target is computed by

YDIST = (COSPHI)(RANGE)

where RANGE is the target range in the TARGETS table.

The distance in range counts that the target moves from ping to ping, LDIST is computed by:

LDIST = (SPEED)(PTIME)(0.190749)

where SPEED is the target speed from the TARGETS table and 0.190749 is a constant relating speed in knots, time in seconds and range counts.

The target course is extracted and converted from degrees to radians. The sine and cosine of the course are computed and stored in SINPH1 and COSPH1 respectively. EXDIST is incremented by the product of LDIST and SINPH1; YDIST is incremented by the product of LDIST and COSPH1. Using FLOP, the new target range is computed by

BEARING = ARCTAN (EXDIST/YDIST)

This target bearing is stored in the appropriate position in the TARGETS table. Note the similarity of the above two equations with equations (17) and (18).

The end of ping mode immediately follows the update targets mode. B2 is checked. If B2 is not equal to PINGS then B2 is incremented, control is transferred to TARGTEST which prints out the new target parameters and then to the beginning of the formatting of a new ping. Otherwise an end of file is written on tape, signifying the end of the formatted data. Control is then transferred to the calling routine as FORMDATA is exited.

Three subroutines are used by FORMDATA to assist it in its function of formulating the output data; PACKITEM, RANDOM and TARGAMPT. PACKITEM packs and thresholds the data into the output table. RANDOM generates a random number for selecting a noise datum. TARGAMPT is a test routine which prints the target parameters.

1. PACKITEM

The purpose of this procedure is to pack one element of data into the output table if it exceeds the L-threshold. Input is B3. Output is the CDATA table.

The CDATA is a 10001 (octal) word table. It is a 4000 (octal) item two words per item horizontal table. The last word of CDATA is a threshold word. The format of the CDATA table is identical to the format of the correlation data generated by the ASP, which is input to the ASP computer program. The item format is described in Reference 1. It is reproduced below:

Word 0:

29	28 27	26	25 20	19 0
M	NOT USED	S	DOPPLER	RANGE

Word 1:

29 18	17 14	13 2	1-0
NOT USED	BEAM	AMPLITUDE	duplicate range

M is the M-threshold. If the amplitude of the element exceeds the value of the M-threshold, MTHRESH, this bit is set. Otherwise it is cleared. Bit 26 of Word 0 contains the second doppler reference field. It is set if the element is on a second doppler reference and cleared if the element is on

a first doppler reference. The doppler field from bits 20-25 gives the doppler in terms of filter number. In the first doppler reference is centered around 37-40 (octal). The range counter is in the RANGE field from 0-19 bits. Each range count corresponds to a range of approximately 0.256 yards. The two lowest bits of range are duplicated in the second word of the item in bits 0 and 1. Beam information is generated in bits 14-17 in Word 1, each number corresponding to a beam filter. Beams are centered between 7 and 10 (octal), 7 corresponding to -1° and 10 corresponding to +1°. The twelve bits of amplitude are generated in bits 2-13 of Word 1. Bit 27 and 28 of Word 0 and 18-29 of Word 1 are not used.

Upon entry to PACKITEM, B3 is divided by DOPPS. The quotient is stored in CBEAM and the remainder in CDOPPLER. The B3rd element of ARRAY is compared to LTHRESH. If it is less than LTHRESH, PACKITEM returns control to the calling subroutine without formatting the element. Otherwise the formatting procedure is initiated. The item doppler is given by the sum of CDOPPLER and DOPDIF. Two items are actually packed for each element of ARRAY that exceeds the L-threshold. They are identical in all respects except the S-field. The first item is on the first doppler reference; the second item is on the second doppler reference. Range and doppler of both items are packed into the B1st and B1st+1 items of CDATA table.

The item beam is given by the sum of CBEAM and BEAMDIF. The B3rd element of ARRAY is checked. If it is greater than full scale amplitude, its value is then set to full scale, 4095. Adjusted beam, amplitude and the two lowest ranges are packed into A and stored in the second word of the CDATA table.

B1 is incremented then tested. If B1 is not zero, it is incremented again and control is transferred back to the calling routine. Otherwise, B1 is cleared, the CDATA table is written on tape, then cleared, and control is transferred back to the calling routine.

2. RANDOM

This procedure generates a random number 28 bits long and enters it in A. RANDOM makes use of two numbers, R1 and R2, to generate the random numbers, 123456 (octal) is entered in Q. If R1 is less than Q, R1 is incremented by Q. If R2 is less than Q, R2 is incremented by Q. R1 is multiplied by R2; the product goes to A and Q. Q is stored in R1. A is incremented by Q and stored in R2. The top two bits of A are cleared. The random number is now in A. RANDOM returns to the calling routine.

There is nothing magic about the test number initially entered in Q. Another value could have been selected. However, a test number of some sort is necessary to prevent R1 and R2 from becoming too low in value. This random number generator is perhaps not the best available nor is it necessarily the fastest. However, an interval test of it over large numbers shows that it has an excellent even distribution property.

3. TARGAMPT

This procedure prints the amplitude and other parameters of a target item. Q is assumed to contain the amplitude and is stored in AMP. CDOPPLER is incremented by DOPDIF to properly center the doppler value. CDOPPLER is converted from filter number to its value in knots and is stored in Z3. The formula is

Z3 = (CDOPPLER-31.5)(DSPREAD)(2)/FF.

B3 is checked. If B3 equals zero then TARGAMPT prints CDOPPLER, Z3, CBEAM and AMP. It then exits the routine. If B3 is not zero, a different set of parameters are printed. The target amplitude test header is printed. SAMPULSE is converted from range in counts to range in miles and stored in Z2. SAMPULSE, Z2, CDOPPLER, Z3, CBEAM and AMP are printed. TARGAMPT is then exited.

TARGAMPT is used to print item parameters, especially amplitude, whenever FORMDATA is in the target generation mode. B3 = 0 indicates the first item of a target sample pulse. All others are subsequent words.

V. SYSTEM OUTPUT

A. SETMTHRESH

This procedure rewrites a whole reel of output tape from tape unit M3 to M4. It examines the data formatted onto M3 as it reads it in and sets the M-threshold of the top 1/64 amplitude items. Output on M4 is in the ASP test format.

NOISTEST is called upon first to test the amplitude distribution of the output data on M3. TOPONE is the number of amplitudes in the NOIDIST table which exceed the L-threshold but which should not exceed the M-threshold. TOPONE is computed by

TOPONE = SUM-SUM/64

where SUM is the total number of amplitudes in NOIDIST.

B1 and RUNSUM are cleared. B1 is used as the NOIDIST index. A search is initiated through the NOIDIST table to determine the level at which to set the M-threshold. The number of items in each word of NOIDIST is summed into RUNSUM until RUNSUM is greater than or equal to TOPONE. When this point is reached, B1 is stored into MTHRESH, M3 is rewound and M4 is positioned to the file designated by FILE. The M-threshold is also stored in the upper half of the last word of the CDATA table.

The RECORDUM table is cleared. RECORDNUM is a 206 word one item per word table. It is designed to be the last record of M4 on each file and it contains information about that file. The first word of RECORDNUM contains the number of data records in the file. Succeeding words of RECORDNUM contain the last range count in the corresponding correlation data record. 55 words of RECORDNUM are allocated for this purpose. The last 150 words would contain the cluster

list after processing by the ASP computer program. SIMULATASP leaves provision for this on magnetic tape, but does not itself do any cluster processing of the data.

The first record of the file, which contains the PINGNO table, is read into core from M3. An end of file here indicates that the end of the data has been reached on M3. The program transfers control to the write end of tape marker mode. If no end of file is read, the PINGNO table is written onto M4.

1 is entered into B1. B1 is used here as an index for RECORDNUM. A CDATA table record is read into core from M3. If an end of file is read at this point, control is transferred to write last record and end of file mode. The range mark is entered in Q and 7777 (octal) in B7. The CDATA table is searched from the end back for the last value in the table. The correlation data range as specified by B7 is extracted and stored in the B1st item of RECORDNUM. B1 is incremented; the first word of RECORDNUM is incremented.

The amplitude mask is entered in Q. B2 is indexed through the CDATA table. The CDATA amplitudes are compared to MTHRESH. The M-threshold bit in each CDATA element is set for each one for which the amplitude is greater than or equal to MTHRESH. When the whole table has been scanned, the CDATA table is written on M4.

When the write last record and end of file mode has been reached, the RECORDNUM table is written on M4 followed by an end of file. The RECORDNUM table is again cleared and another ping is transferred from M3 to M4 as described above.

When the write end of tape marker has been reached, an end of file is written on M4. Both tapes are rewound. Control is transferred to the calling routine.

B. NOISTEST

This procedure tests the noise amplitude distribution of the output data. It analyzes the first ping of data formatted and prints the amplitude distribution tables.

Upon entry M3 is rewound. M3 is then spaced one record forward.

The NOIDIST table is cleared. NOIDIST is a 10000 (octal) word one word per item table. Its name stands for noise amplitude distribution test table. It contains, after the analysis of one ping of data, the number of data in each amplitude bin on a whole ping from 0 to full scale. SUM is cleared. SUM contains the total number of data in the ping.

The following sequence is repeated until an end of file is read from M3 or a zero amplitude is reached. One record of data is read into the CDATA table. B1 is indexed through the table. The CDATA amplitude is entered into B7. The B7th item of NOIDIST is incremented and SUM is incremented.

After the NOIDIST table has been filled, RUNSUM is cleared. A top of form command is sent to the printer. The output noise test header and the cumulative amplitude probability header are printed. The index and columns header is printed. B1, B2, and B3 are cleared. The following sequence is repeated line by line until the whole cumulative amplitude probability noise distribution table has been printed; in this sequence B1 is the line counter, B2 is the NOIDIST index, and B3 is the PARAM index.

B2 is stored in Z1. RUNSUM is incremented by the B2nd item of NOIDIST. RUNSUM is divided by SUM; the quotient is stored in the B3rd item of PARAM. B2 is incremented. This sequence is repeated 10 times,

indexing B3 through PARAM. The AOP table is cleared. The 10 PARAM values are converted from fixed point to floating and stored in AOP. This process sets up one line of 10 values to be printed on the printer. The 10 AOP values are printed along with their index, Z1.

The above sequence is repeated 410 times as B1 is indexed from 0 through 409. The first table has been printed at this point.

A top of form is generated to initiate a print of the second table. The following three headers are printed, in order: the output noise test header, the probability density function header, and the index and columns header, B2 is cleared. The following sequence is repeated until the entire NOIDIST table has been printed.

B2 is stored in Z1. Ten cells are moved from NOIDIST+B2 to AOP. B2 is incremented by 10. One data line is printed; the index, Z1, plus the 10 AOP's. This sequence is repeated 410 times, after which control is returned to the calling routine.

VI. SPECIAL TEST ROUTINES

In addition to the other output routines, there are two special output routines. They are AMPTEST and RANDOMTEST which are selectable by manual key settings. Setting key 1 selects AMPTEST and setting key 2 selects RANDOMTEST.

A. AMPTEST

This procedure is similar to NOISTEST and it prints two tables which are almost identical in format to the tables printed out by NOISTEST. NOISTEST prints the values in the NOIDIST table whereas AMPTEST prints out the values in the NOIPROB table. As mentioned before, NOIPROB contains the cumulative noise amplitude distribution probability. If key 1 is set AMPTEST is called upon immediately after FORMNOISE.

B. RANDOMTEST

This procedure tests the randomness of the distribution of random numbers generated by the RANDOM S/R. A 1000 (octal) word table is cleared. RANDOM is called upon 262,144 times. Each value generated is scaled down to 9 bits. The appropriate item in the table is incremented. Then the values of the table are printed, ten to a line. The basic idea here is that if the values are fairly evenly distributed, RANDOM is a good generator for our purposes. In fact when RANDOM was so tested, the distribution was evenly distributed in a random manner.

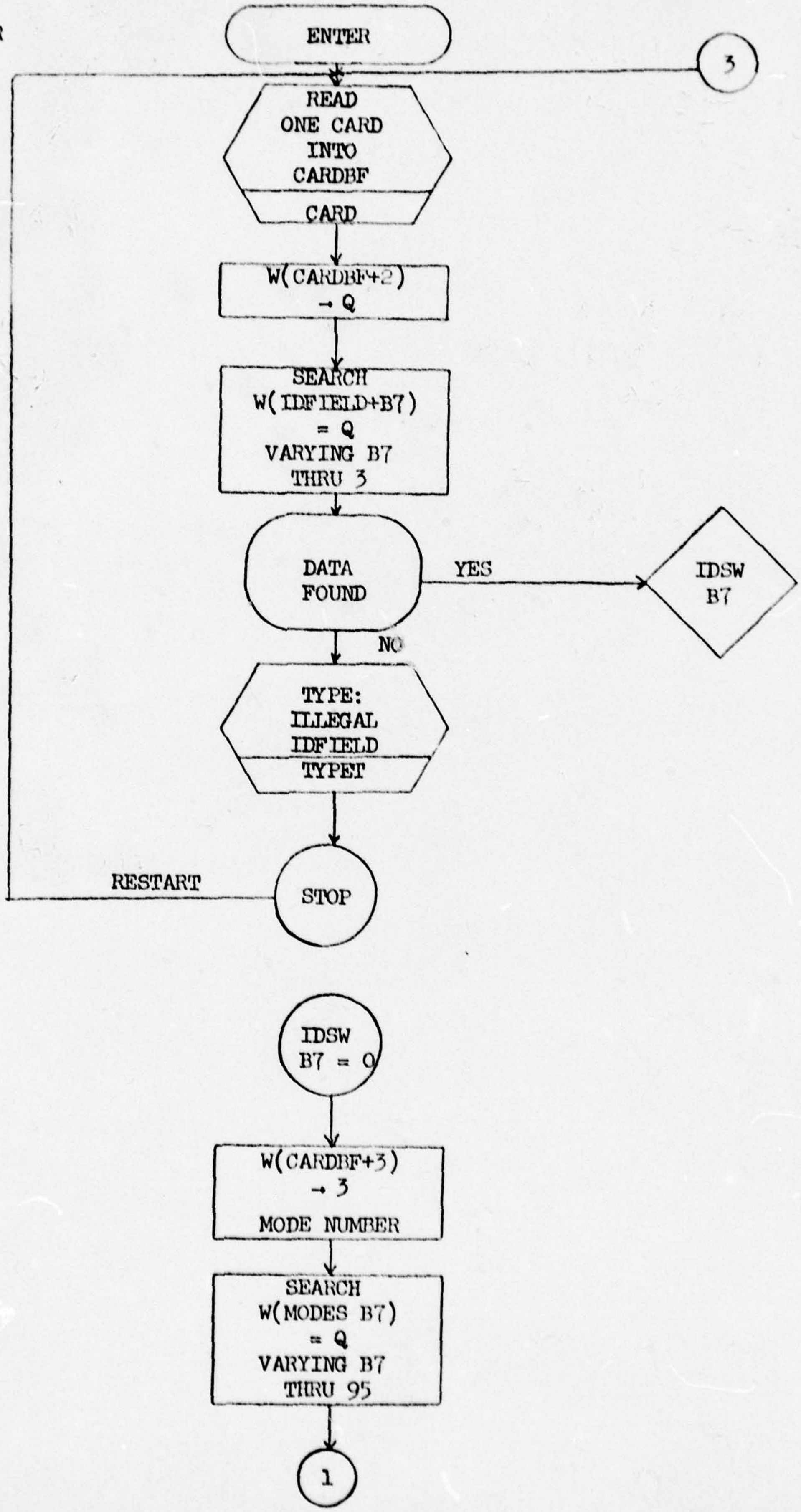
REFERENCES

1. "Specification for an Active Sonar Processor System", NEL Code 3150, 30 June 1967.

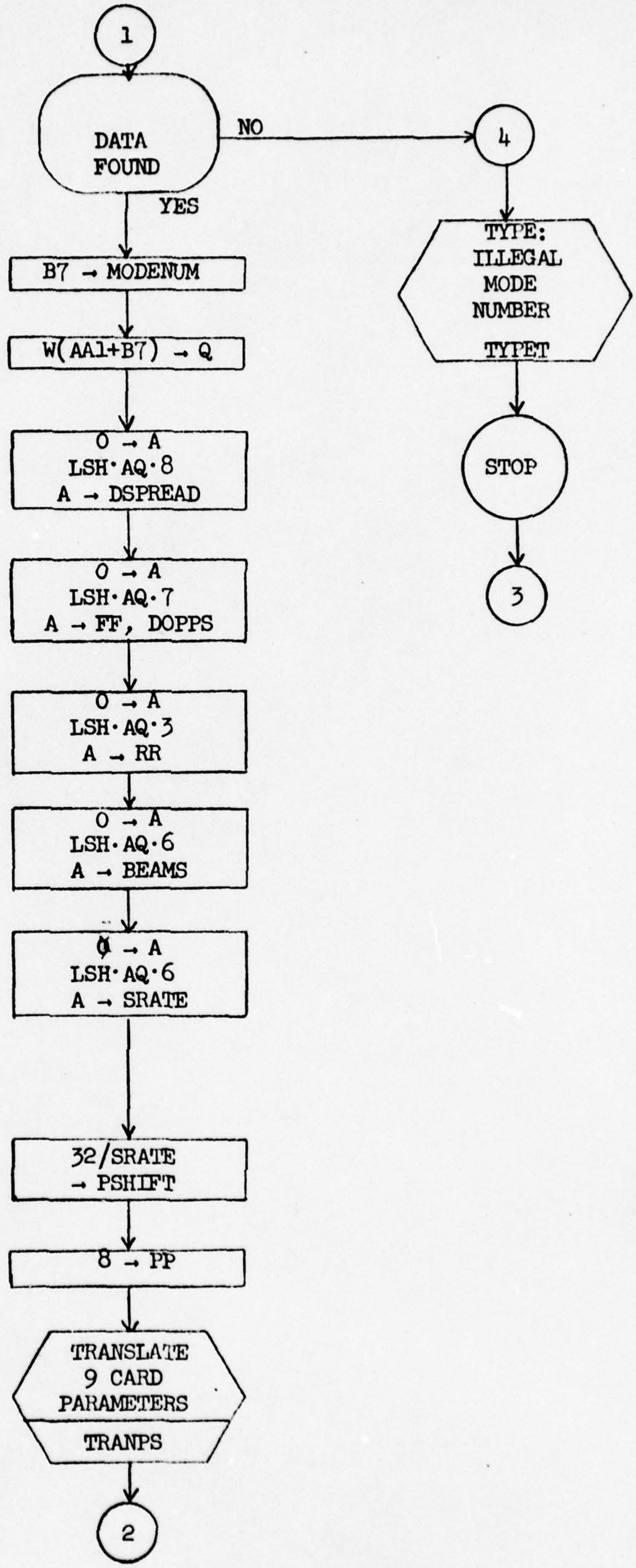
APPENDIX

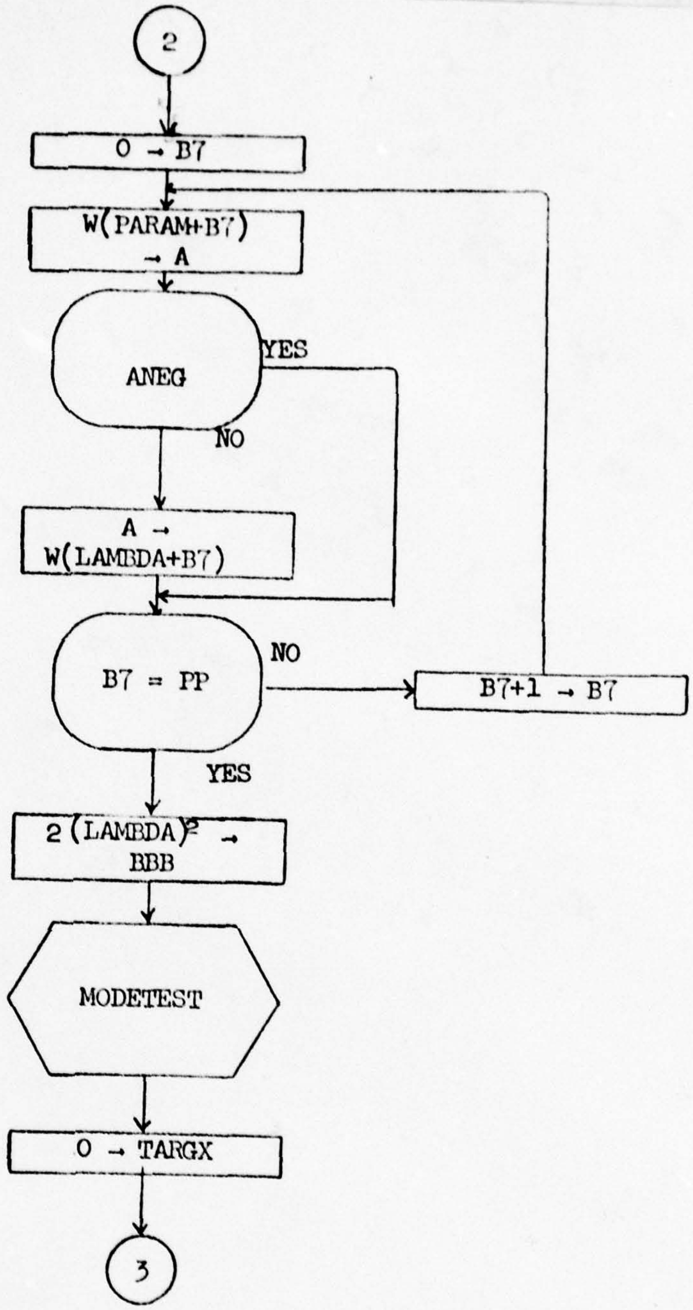
Flow Charts

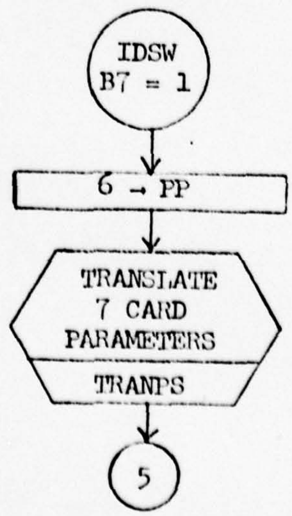
CARDINPUT S/R

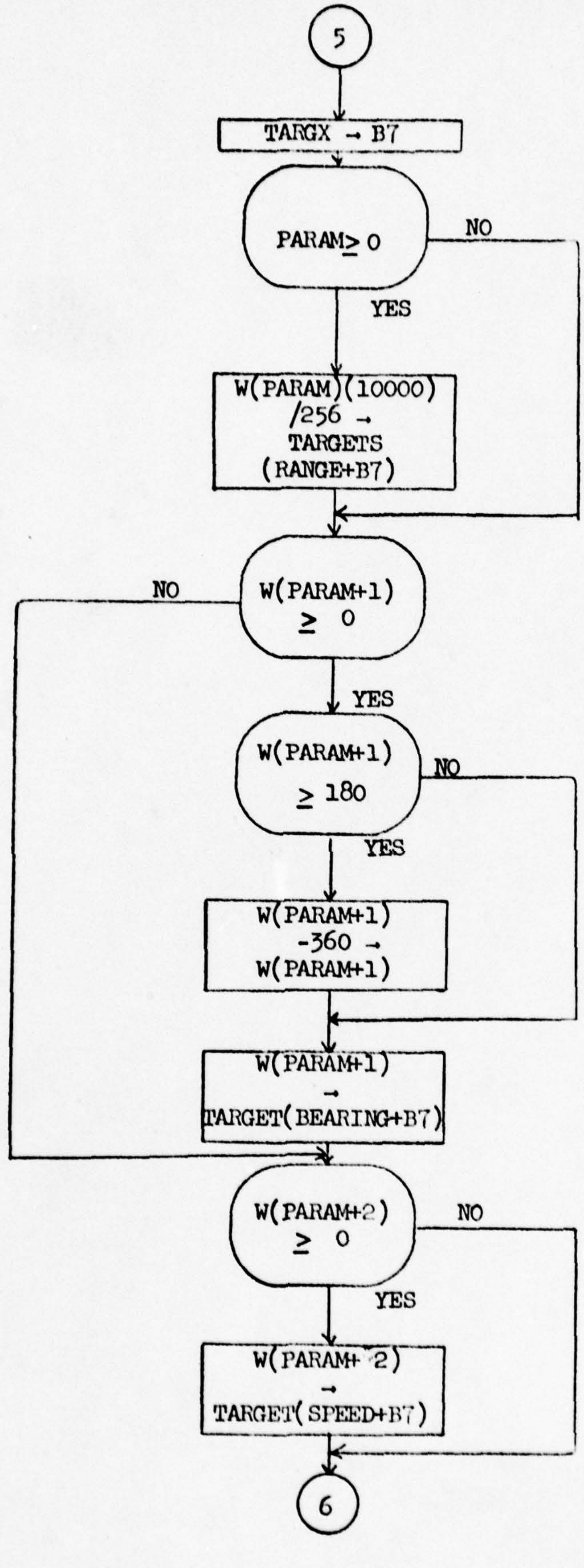


CARDINPUT S/R
PAGE 2

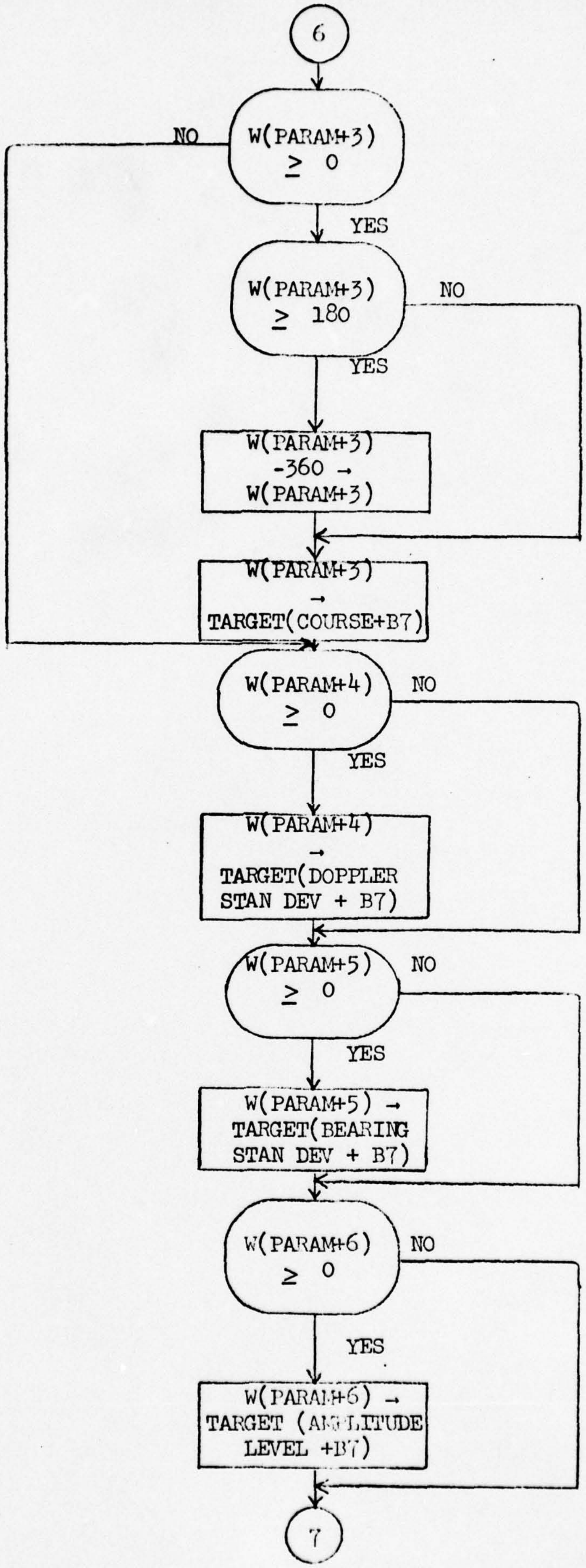


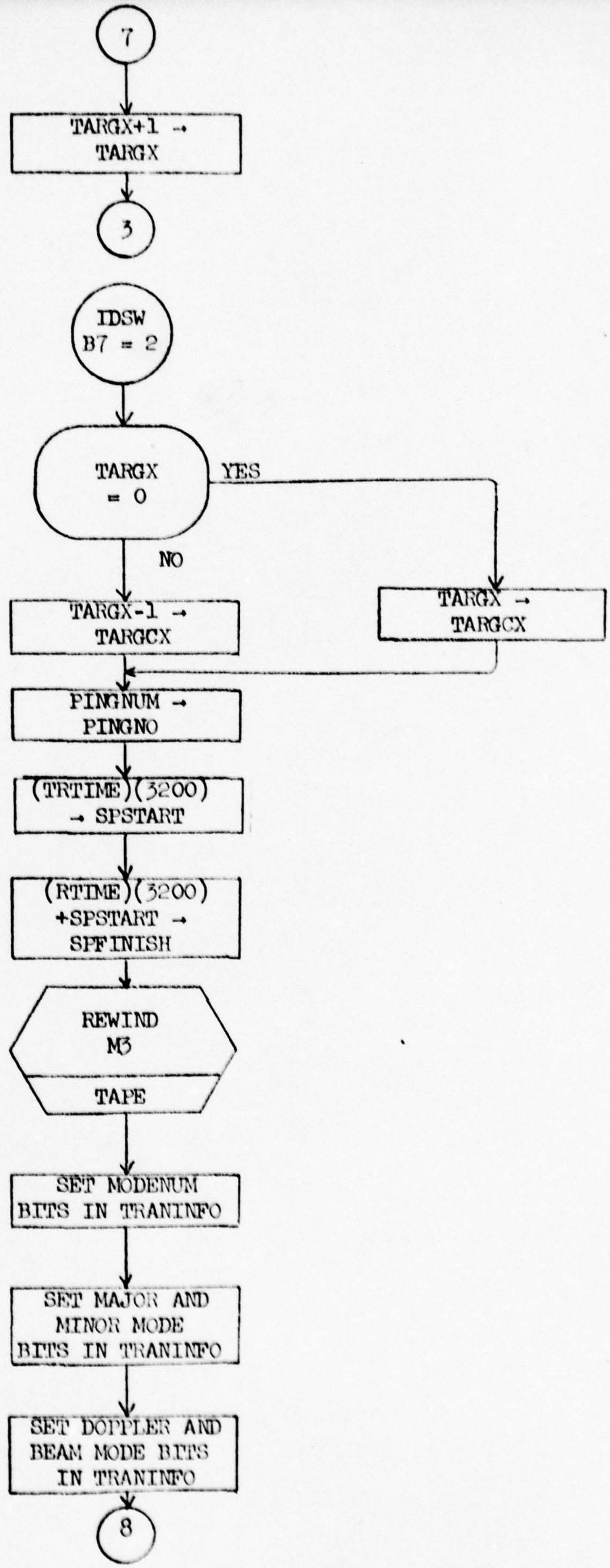

2
0 → B7
W(PARAM+B7)
→ A
ANEG
YES
NO
A →
W(LAMBDA+B7)
B7 = PP
NO
B7+1 → B7
YES
2 (LAMBDA)² →
BBB
MODETEST
0 → TARGX
3

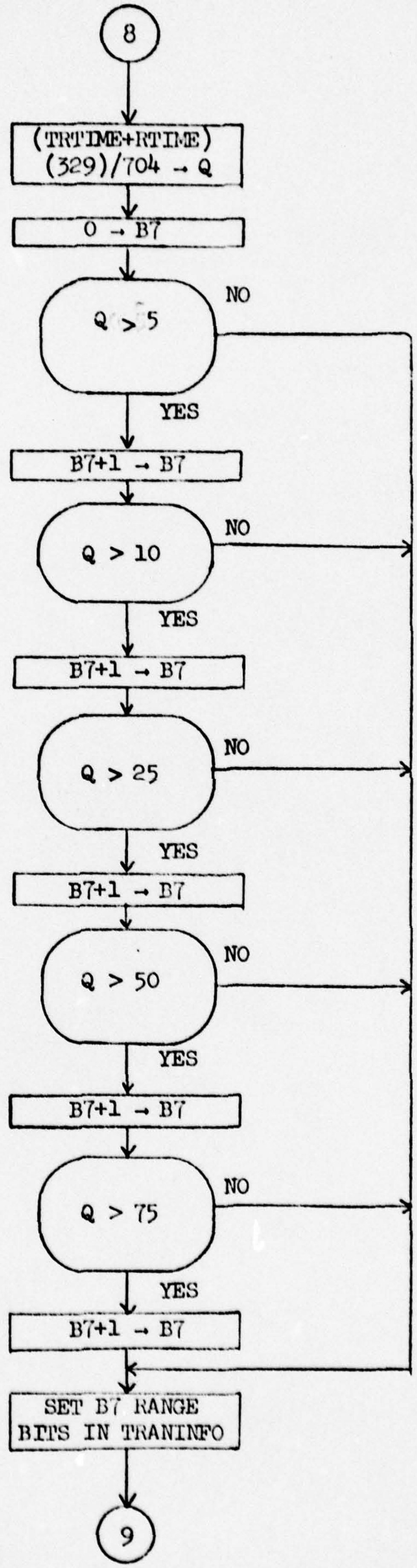

IDSW
B7 = 1
6 → PP
TRANSLATE
7 CARD
PARAMETERS
TRANPS
5

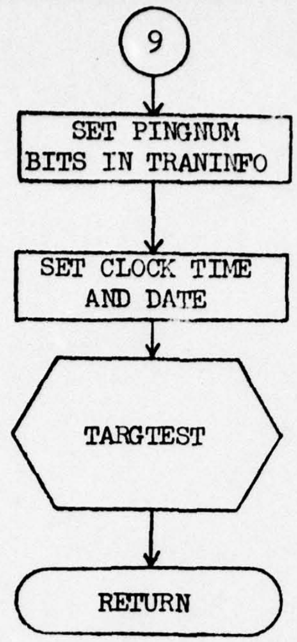

5
TARGX → B7
PARAM ≥ 0
NO
YES
W(PARAM)(10000) /256 → TARGETS (RANGE+B7)
W(PARAM+1) ≥ 0
NO
YES
W(PARAM+1) ≥ 180
NO
YES
W(PARAM+1) -360 → W(PARAM+1)
W(PARAM+1) → TARGET(BEARING+B7)
W(PARAM+2) ≥ 0
NO
YES
W(PARAM+2) → TARGET(SPEED+B7)
6

CARDINPUT S/R
PAGE 5

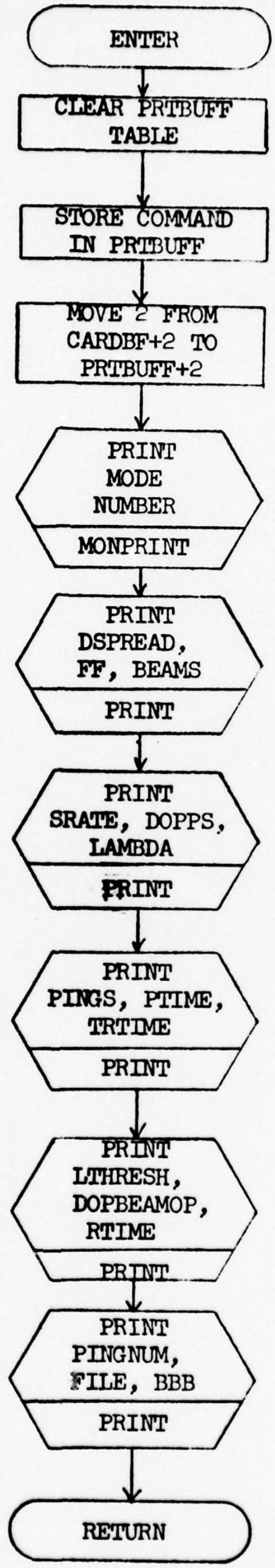



7
TARGX+1 → TARGX
3
IDSW B7 = 2
TARGX = 0
YES
NO
TARGX-1 → TARGCX
TARGX → TARGCX
PINGNUM → PINGNO
(TRTIME)(3200) → SPSTART
(RTIME)(3200) +SPSTART → SPFINISH
REWIND M3
TAPE
SET MODENUM BITS IN TRANINFO
SET MAJOR AND MINOR MODE BITS IN TRANINFO
SET DOPPLER AND BEAM MODE BITS IN TRANINFO
8

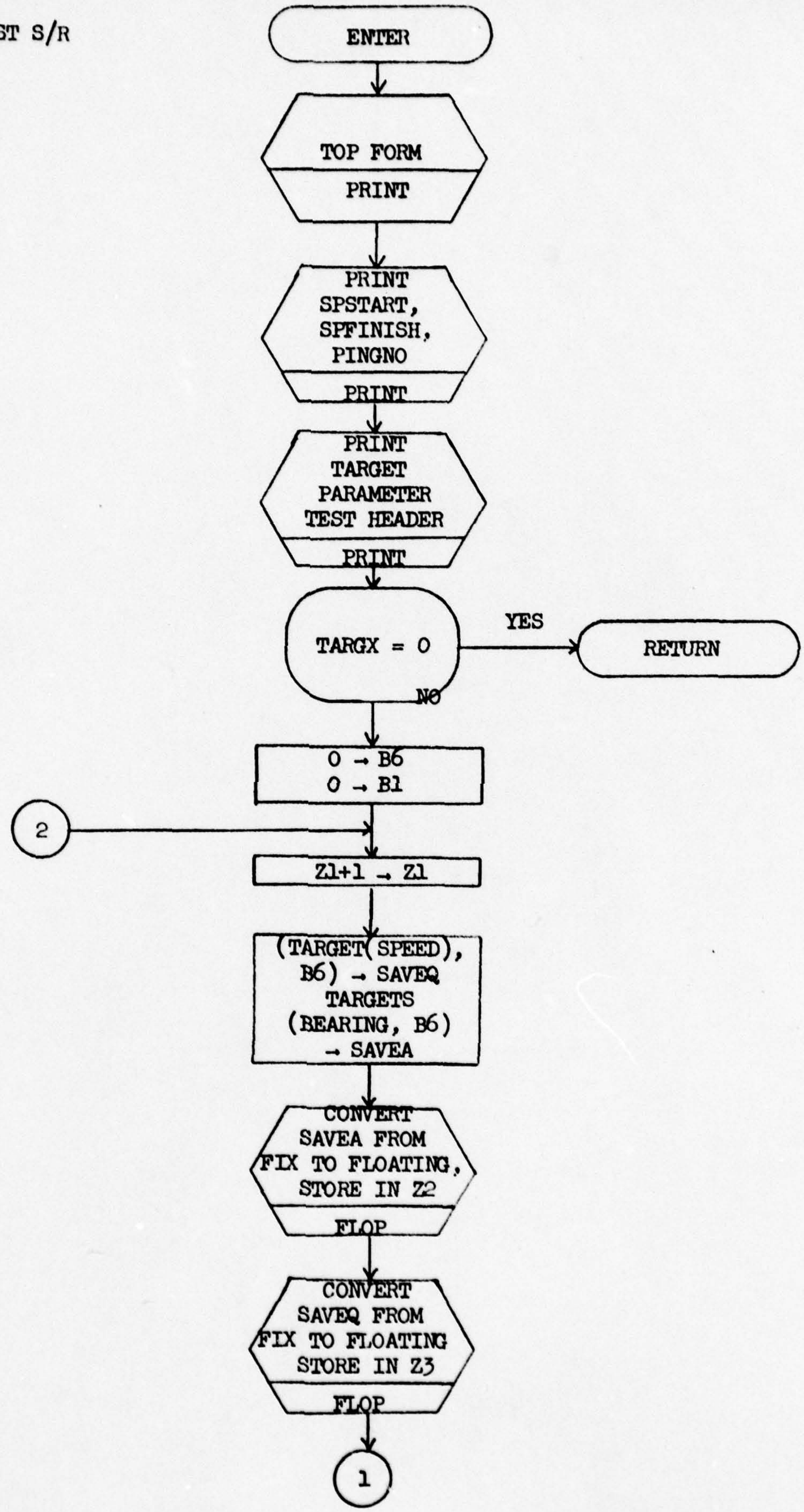

8
(TRTIME+RTIME)
(329)/704 → Q
0 → B7
Q > 5
NO
YES
B7+1 → B7
Q > 10
NO
YES
B7+1 → B7
Q > 25
NO
YES
B7+1 → B7
Q > 50
NO
YES
B7+1 → B7
Q > 75
NO
YES
B7+1 → B7
SET B7 RANGE
BITS IN TRANINFO
9

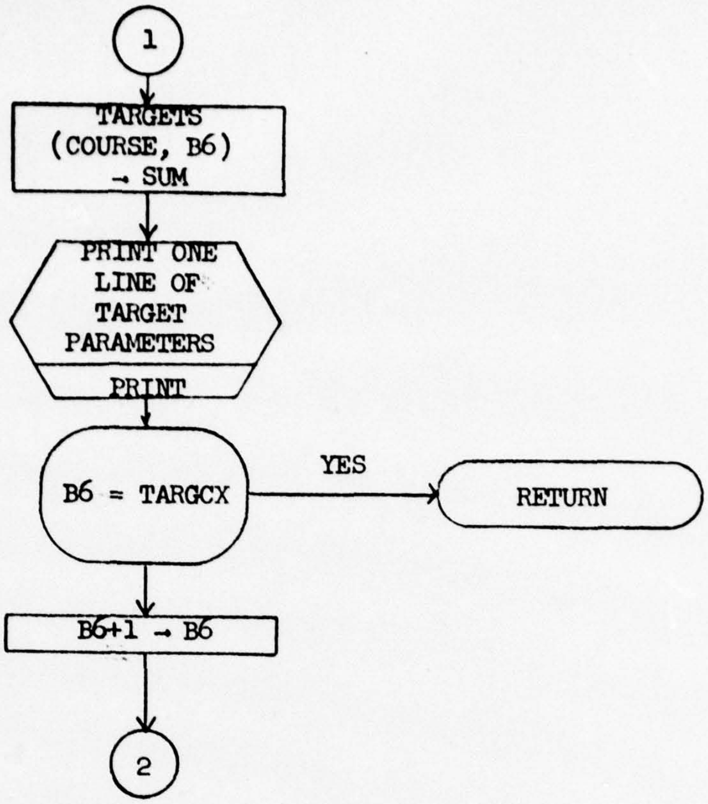

9
SET PINGNUM BITS IN TRANINFO
SET CLOCK TIME AND DATE
TARGTEST
RETURN

MODETEST S/R

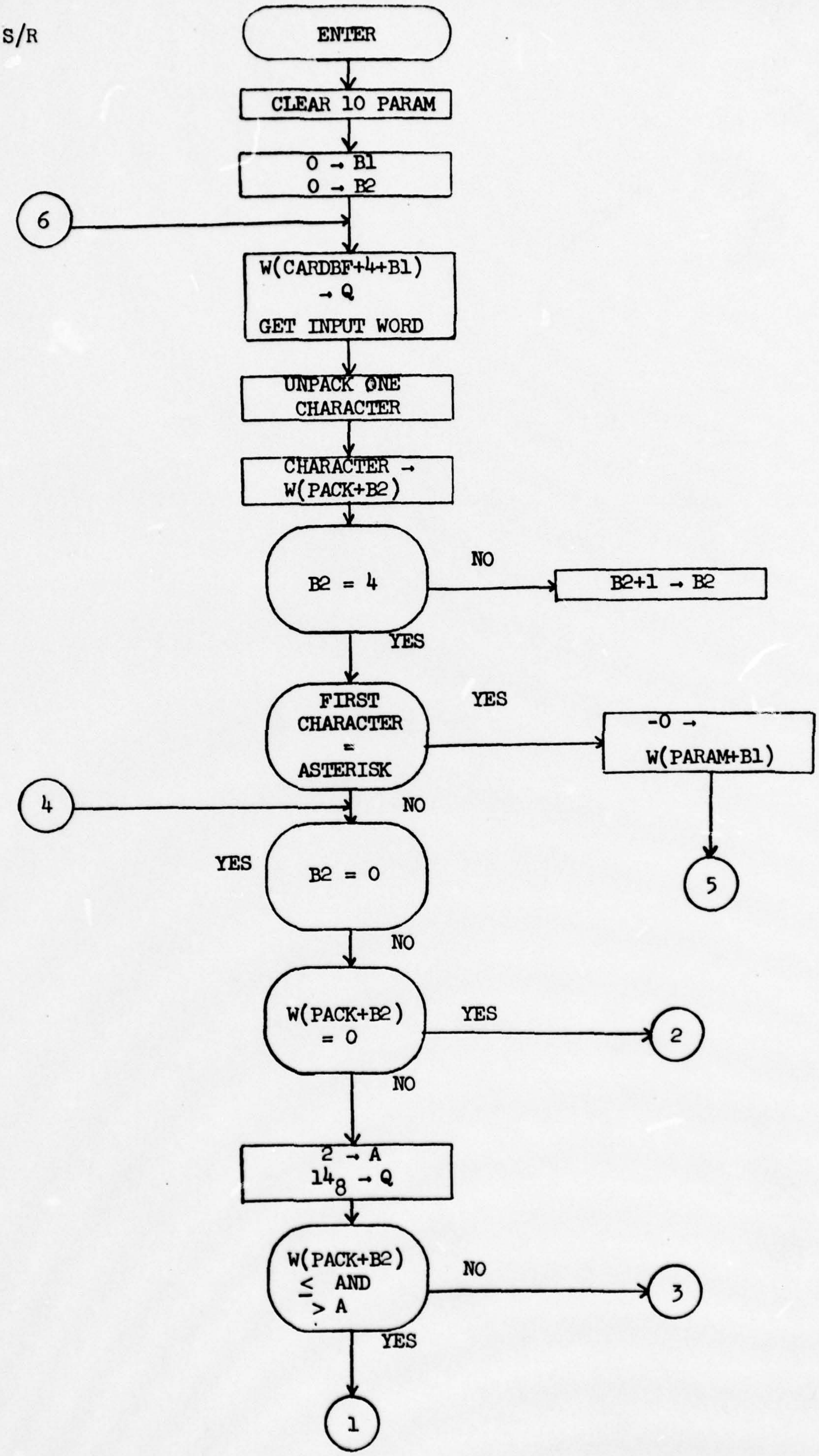

ENTER
CLEAR PRTBUFF TABLE
STORE COMMAND IN PRTBUFF
MOVE 2 FROM CARDBF+2 TO PRTBUFF+2
PRINT MODE NUMBER
MONPRINT
PRINT DSPREAD, FF, BEAMS
PRINT
PRINT SRATE, DOPPS, LAMBDA
PRINT
PRINT PINGS, PTIME, TRTIME
PRINT
PRINT LTHRESH, DOPBEAMOP, RTIME
PRINT
PRINT PINGNUM, FILE, BBB
PRINT
RETURN

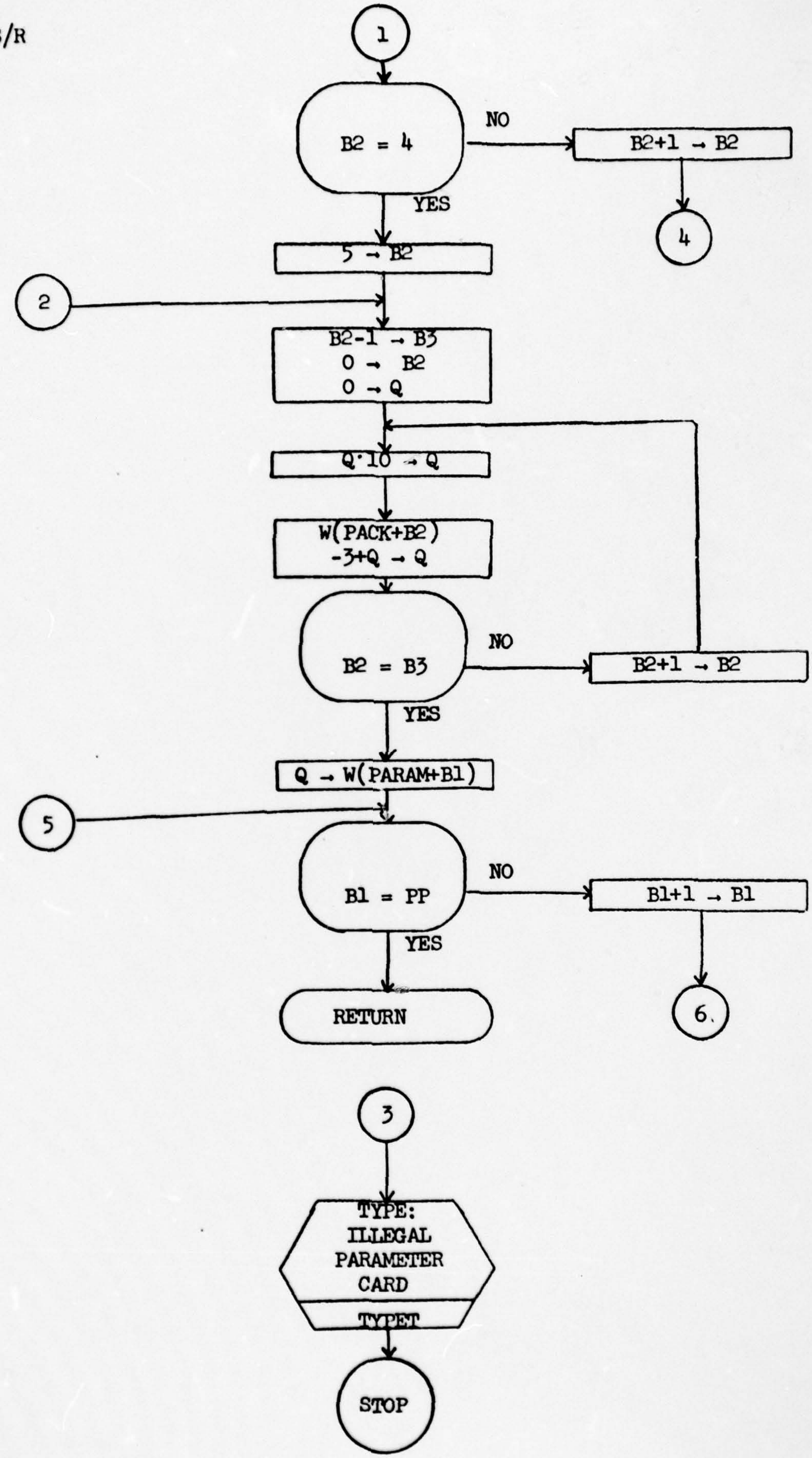
TARGTEST S/R


ENTER
TOP FORM
PRINT
PRINT SPSTART, SPFINISH, PINGNO
PRINT
PRINT TARGET PARAMETER TEST HEADER
PRINT
TARGX = 0
YES
RETURN
NO
0 → B6
0 → B1
2
Z1+1 → Z1
(TARGET(SPEED), B6) → SAVEQ
TARGETS (BEARING, B6) → SAVEA
CONVERT SAVEA FROM FIX TO FLOATING, STORE IN Z2
FLOP
CONVERT SAVEQ FROM FIX TO FLOATING STORE IN Z3
FLOP
1

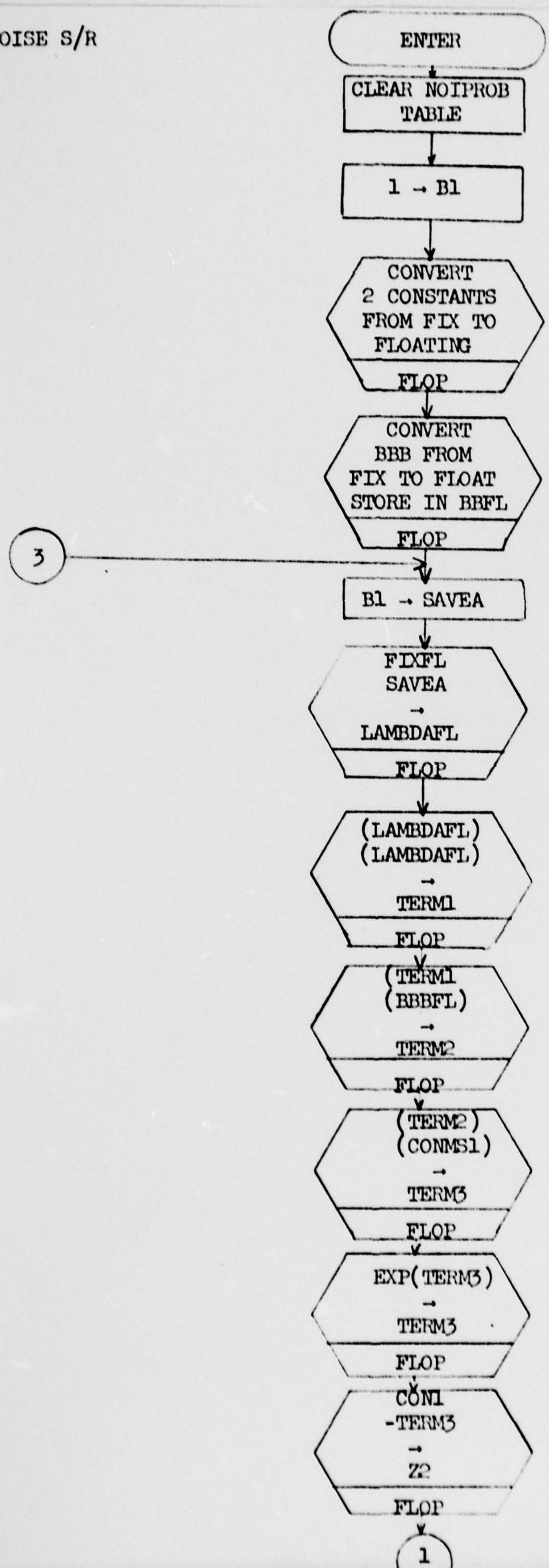

1
TARGETS (COURSE, B6) → SUM
PRINT ONE LINE OF TARGET PARAMETERS
PRINT
B6 = TARGCX
YES
RETURN
B6+1 → B6
2

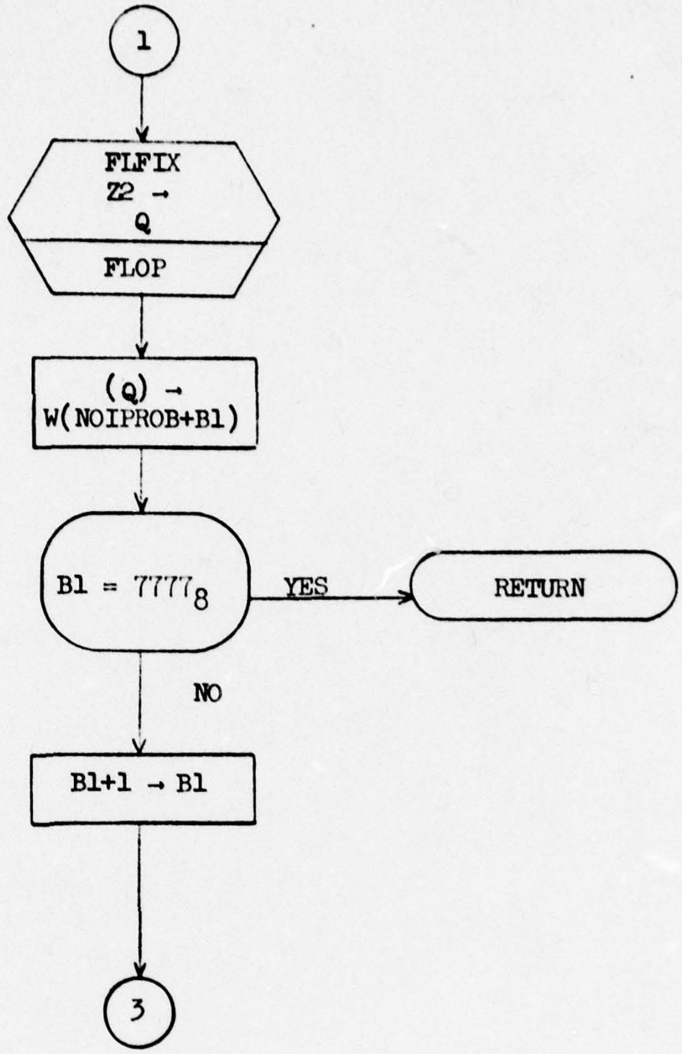
TRANPS S/R


ENTER
CLEAR 10 PARAM
0 → B1
0 → B2
6
W(CARDBF+4+B1)
→ Q
GET INPUT WORD
UNPACK ONE
CHARACTER
CHARACTER →
W(PACK+B2)
B2 = 4
NO
B2+1 → B2
YES
FIRST
CHARACTER
=
ASTERISK
YES
-0 →
W(PARAM+B1)
5
4
NO
YES
B2 = 0
NO
W(PACK+B2)
= 0
YES
2
NO
2 → A
14_8 → Q
W(PACK+B2)
≤ AND
> A
NO
3
YES
1

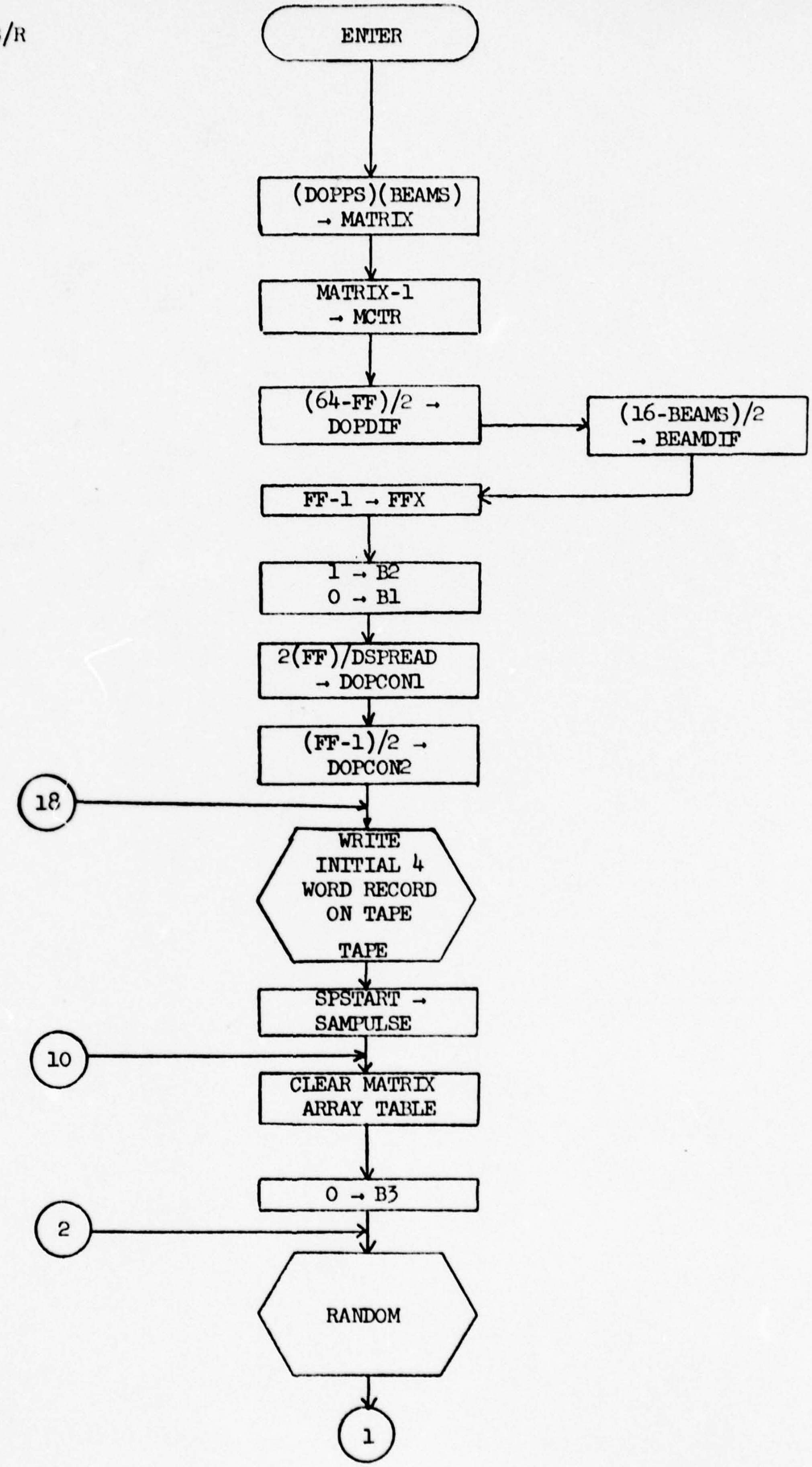

1
B2 = 4
NO
B2+1 → B2
4
YES
5 → B2
2
B2-1 → B3
0 → B2
0 → Q
Q·10 → Q
W(PACK+B2)
-3+Q → Q
B2 = B3
NO
B2+1 → B2
YES
Q → W(PARAM+B1)
5
B1 = PP
NO
B1+1 → B1
YES
RETURN
6.
3
TYPE:
ILLEGAL
PARAMETER
CARD
TYPET
STOP

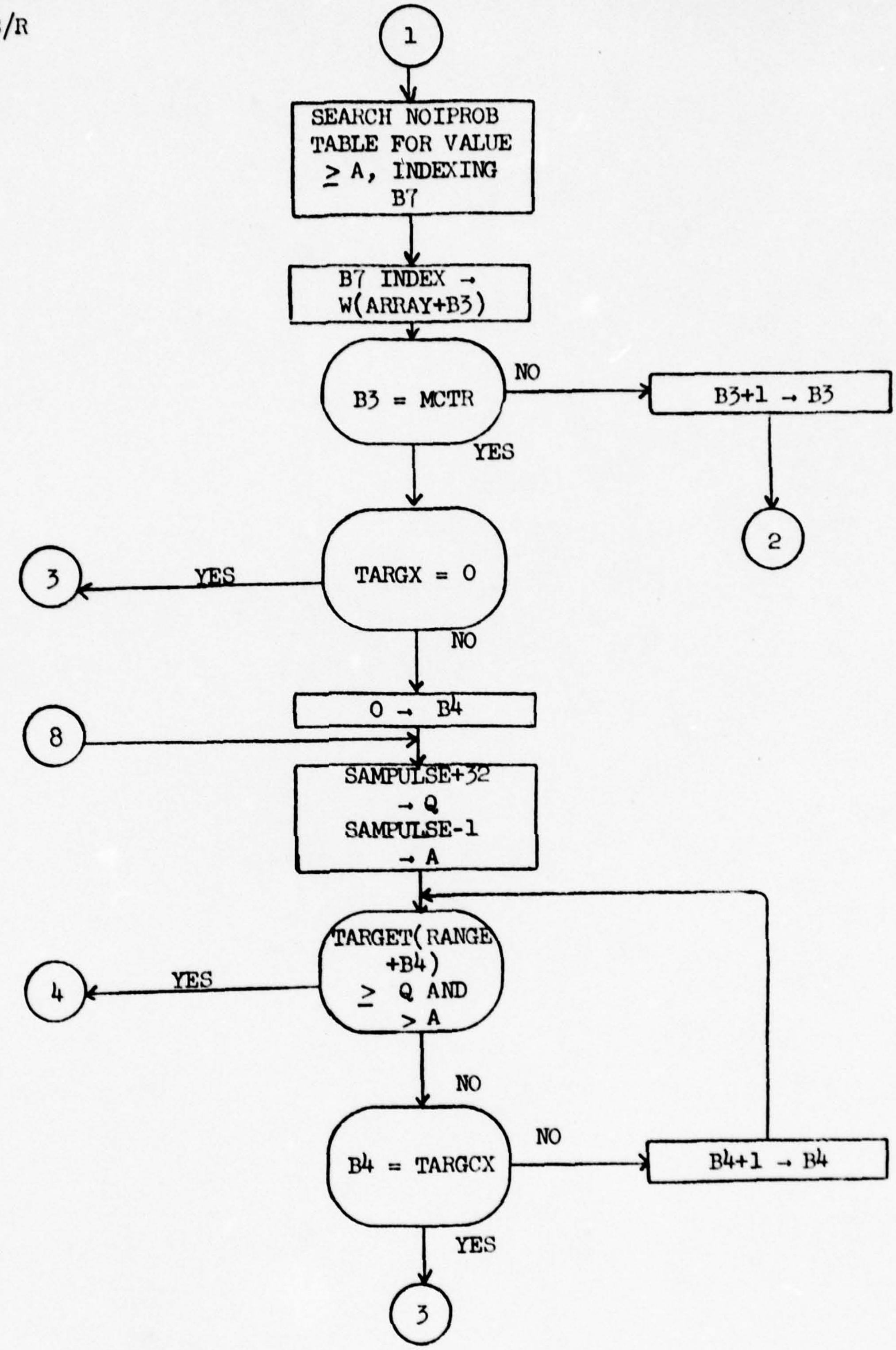
FORMNOISE S/R

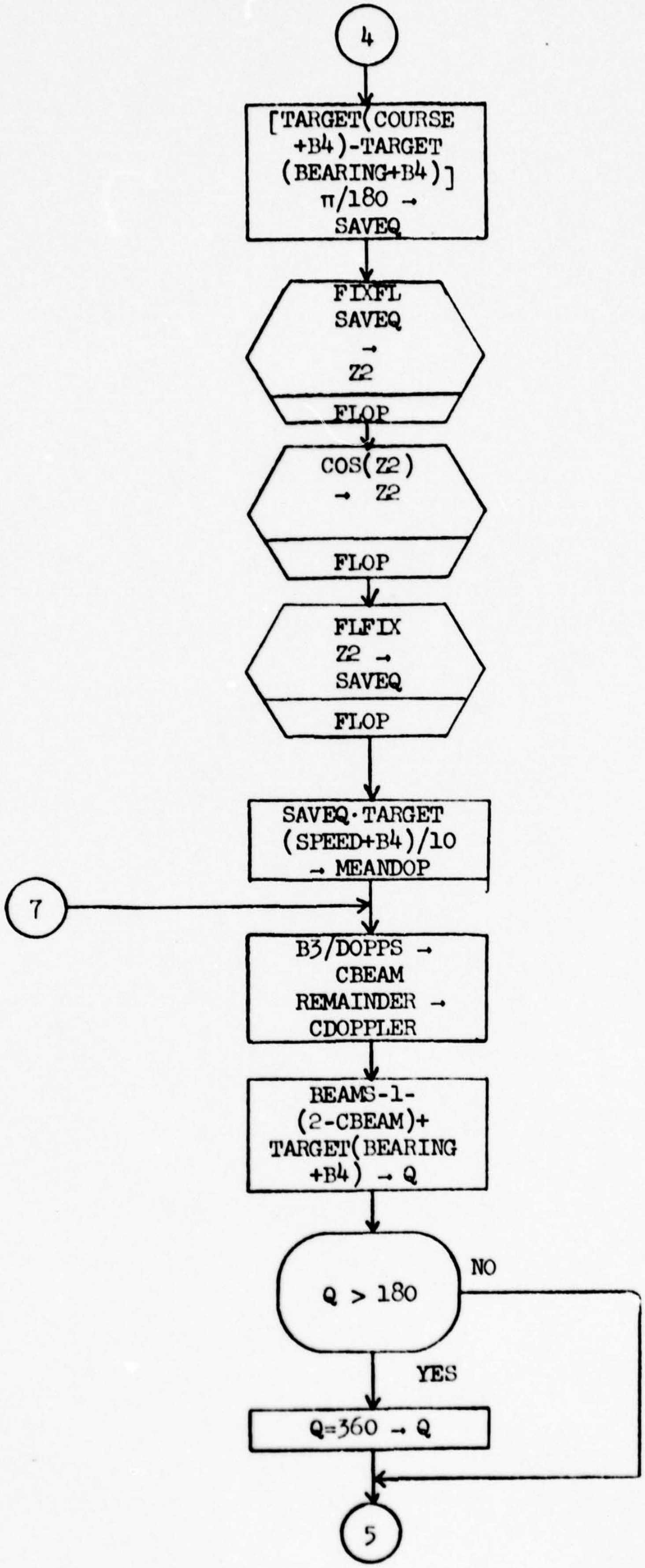



1
FLFIX
Z2 →
Q
FLOP
(Q) →
W(NOIPROB+B1)
B1 = 7777_8
YES
RETURN
NO
B1+1 → B1
3

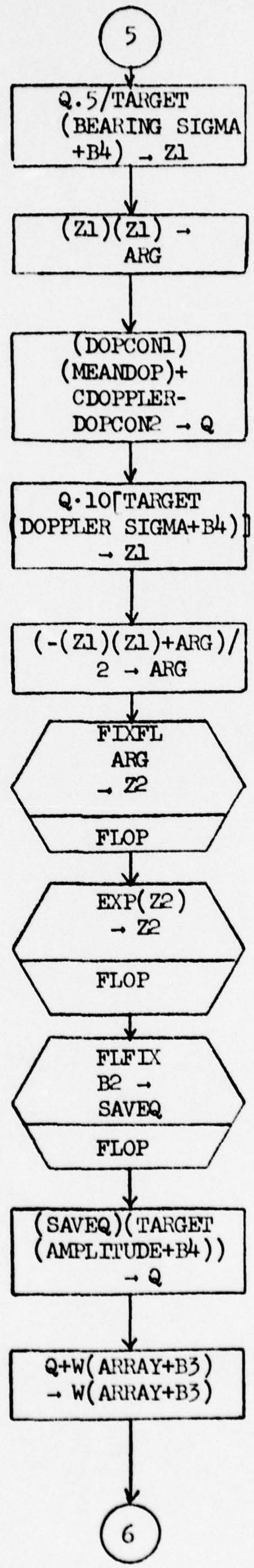
FORMDATA S/R

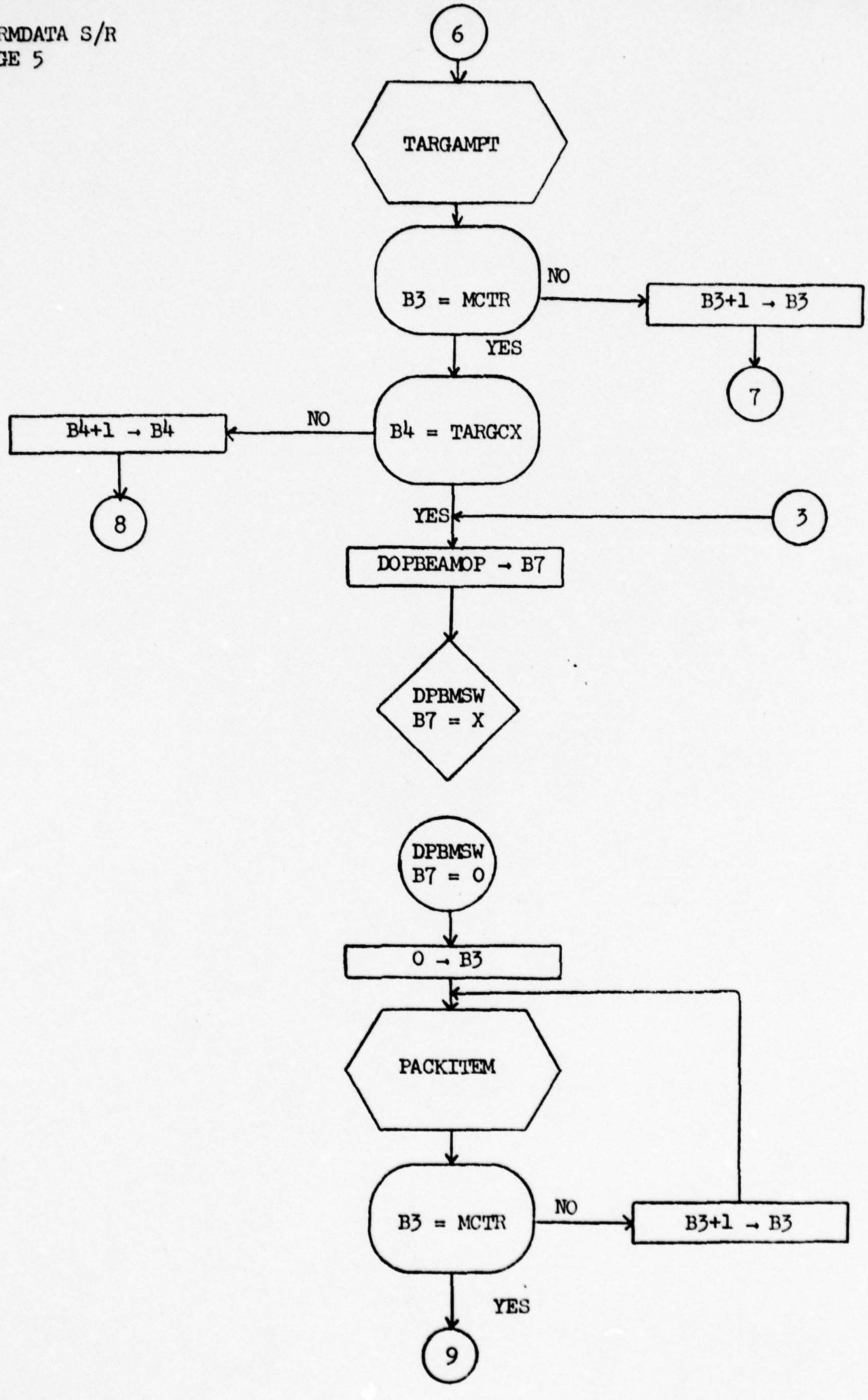

ENTER
(DOPPS)(BEAMS)
→ MATRIX
MATRIX-1
→ MCTR
(64-FF)/2 →
DOPDIF
(16-BEAMS)/2
→ BEAMDIF
FF-1 → FFX
1 → B2
0 → B1
2(FF)/DSPREAD
→ DOPCON1
(FF-1)/2 →
DOPCON2
18
WRITE
INITIAL 4
WORD RECORD
ON TAPE
TAPE
SPSTART →
SAMPULSE
10
CLEAR MATRIX
ARRAY TABLE
0 → B3
2
RANDOM
1


1
SEARCH NOIPROB TABLE FOR VALUE ≥ A, INDEXING B7
B7 INDEX → W(ARRAY+B3)
B3 = MCTR
NO
B3+1 → B3
2
YES
TARGX = 0
YES
3
NO
0 → B4
8
SAMPULSE+32 → Q
SAMPULSE-1 → A
TARGET(RANGE +B4) ≥ Q AND > A
YES
4
NO
B4 = TARGCX
NO
B4+1 → B4
YES
3

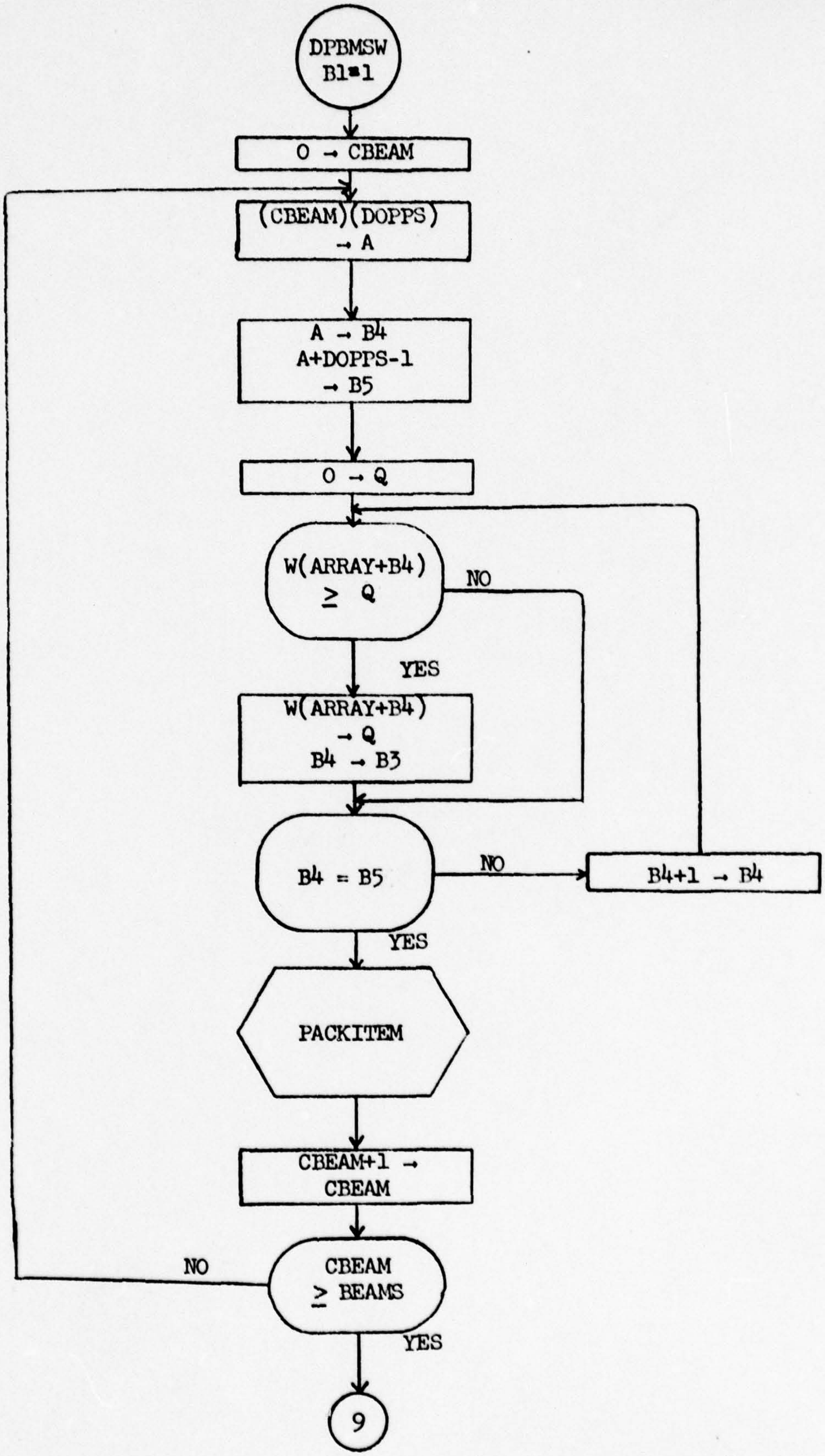

4
[TARGET(COURSE +B4)-TARGET (BEARING+B4)] π/180 → SAVEQ
FIXFL SAVEQ → Z2
FLOP
COS(Z2) → Z2
FLOP
FLFIX Z2 → SAVEQ
FLOP
SAVEQ·TARGET (SPEED+B4)/10 → MEANDOP
7
B3/DOPPS → CBEAM REMAINDER → CDOPPLER
BEAMS-1-(2-CBEAM)+ TARGET(BEARING +B4) → Q
Q > 180
NO
YES
Q=360 → Q
5

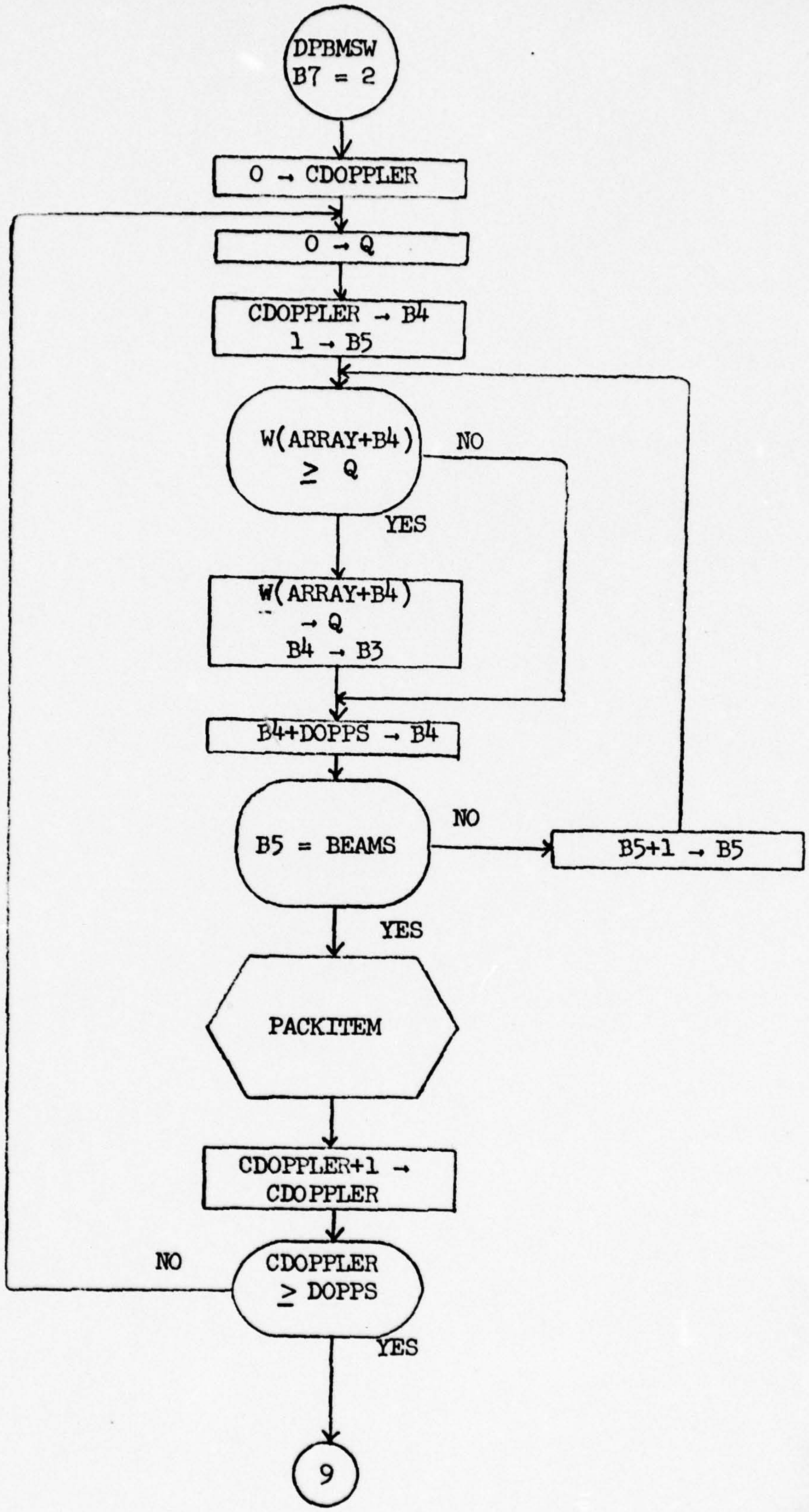
FORMDATA S/R
PAGE 4



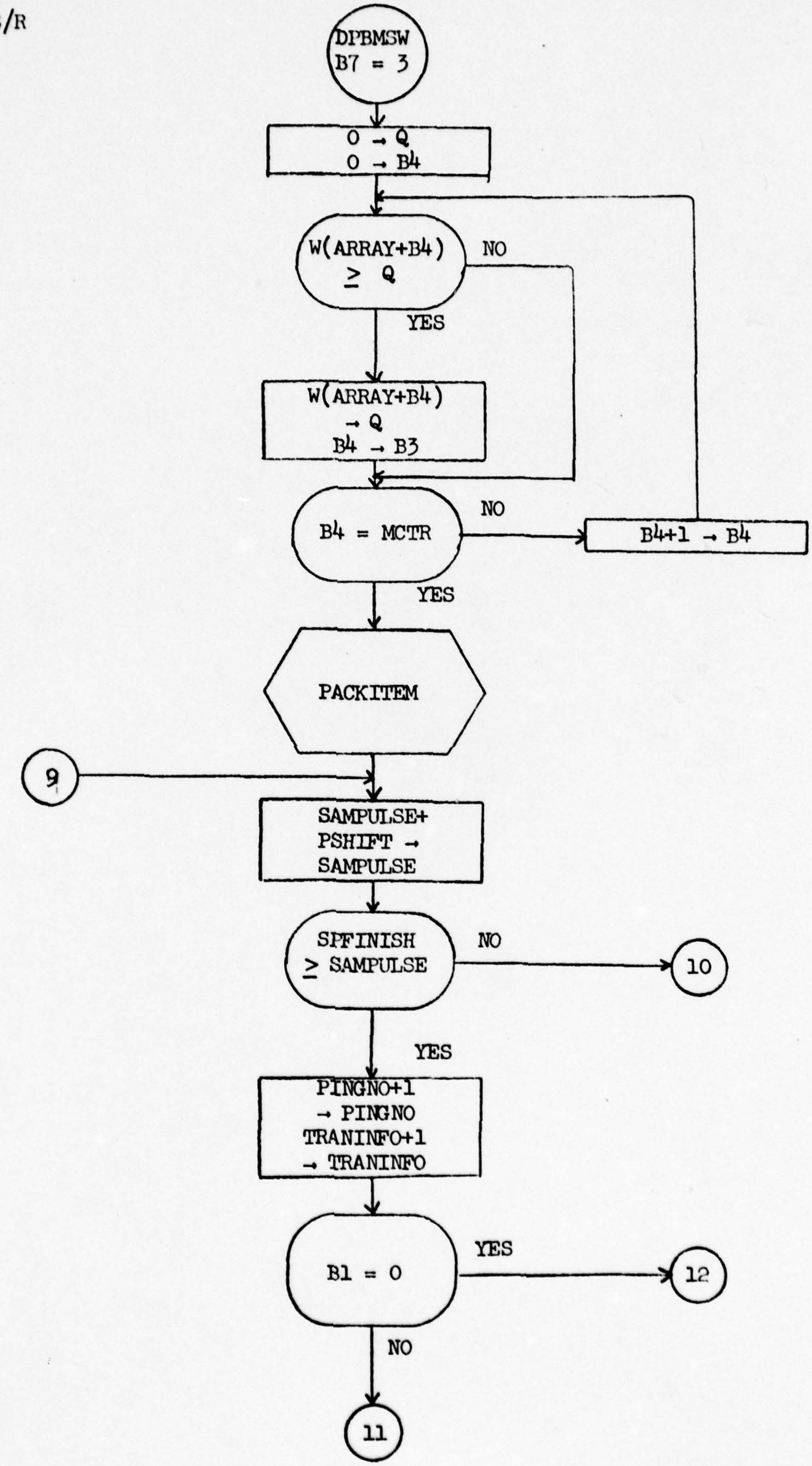

6
TARGAMPT
B3 = MCTR
NO
B3+1 → B3
YES
7
B4 = TARGCX
NO
B4+1 → B4
8
YES
3
DOPBEAMOP → B7
DPBMSW
B7 = X
DPBMSW
B7 = 0
0 → B3
PACKITEM
B3 = MCTR
NO
B3+1 → B3
YES
9

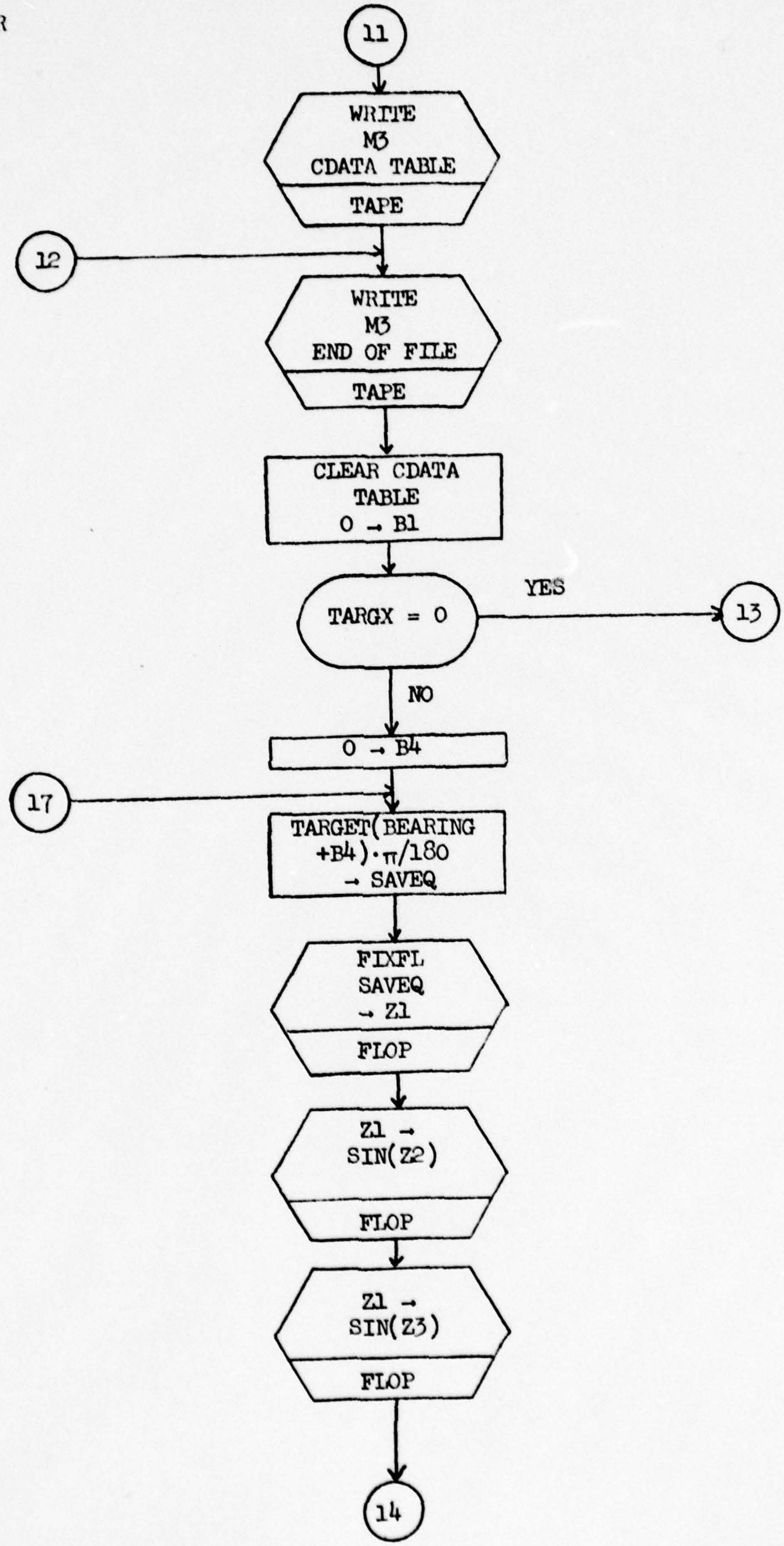
FORMDATA S/R
PAGE 6

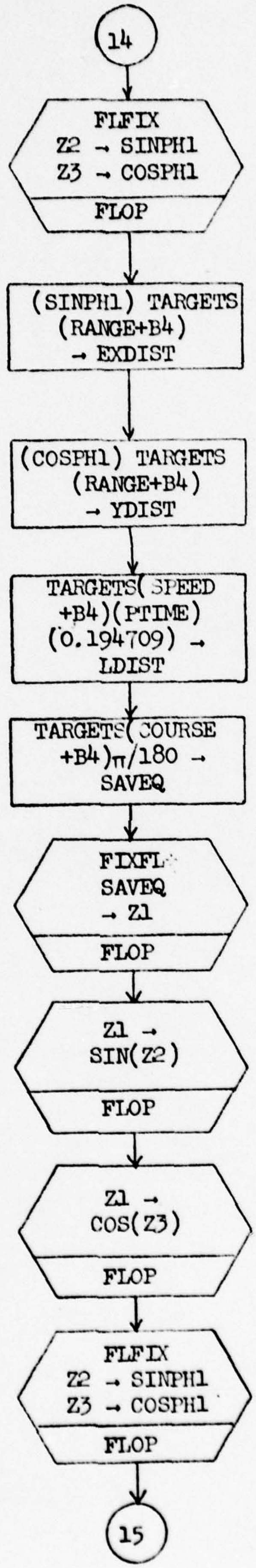


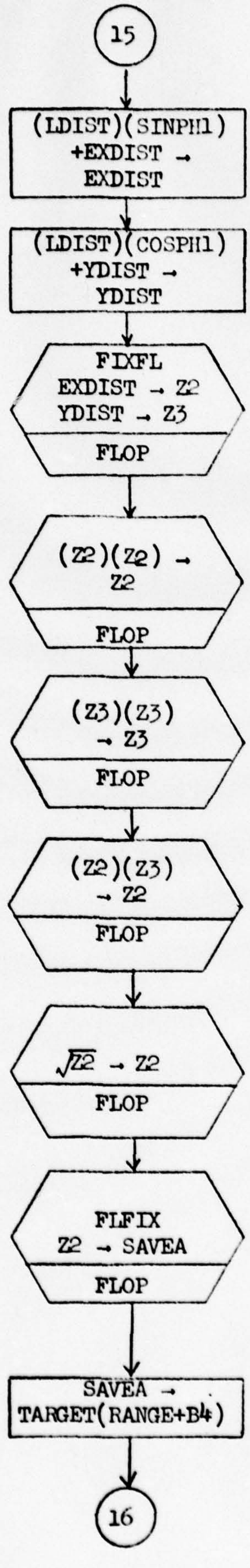

DPBMSW
B7 = 2
0 → CDOPPLER
0 → Q
CDOPPLER → B4
1 → B5
W(ARRAY+B4)
≥ Q
NO
YES
W(ARRAY+B4)
→ Q
B4 → B3
B4+DOPPS → B4
B5 = BEAMS
NO
B5+1 → B5
YES
PACKITEM
CDOPPLER+1 →
CDOPPLER
NO
CDOPPLER
≥ DOPPS
YES
9

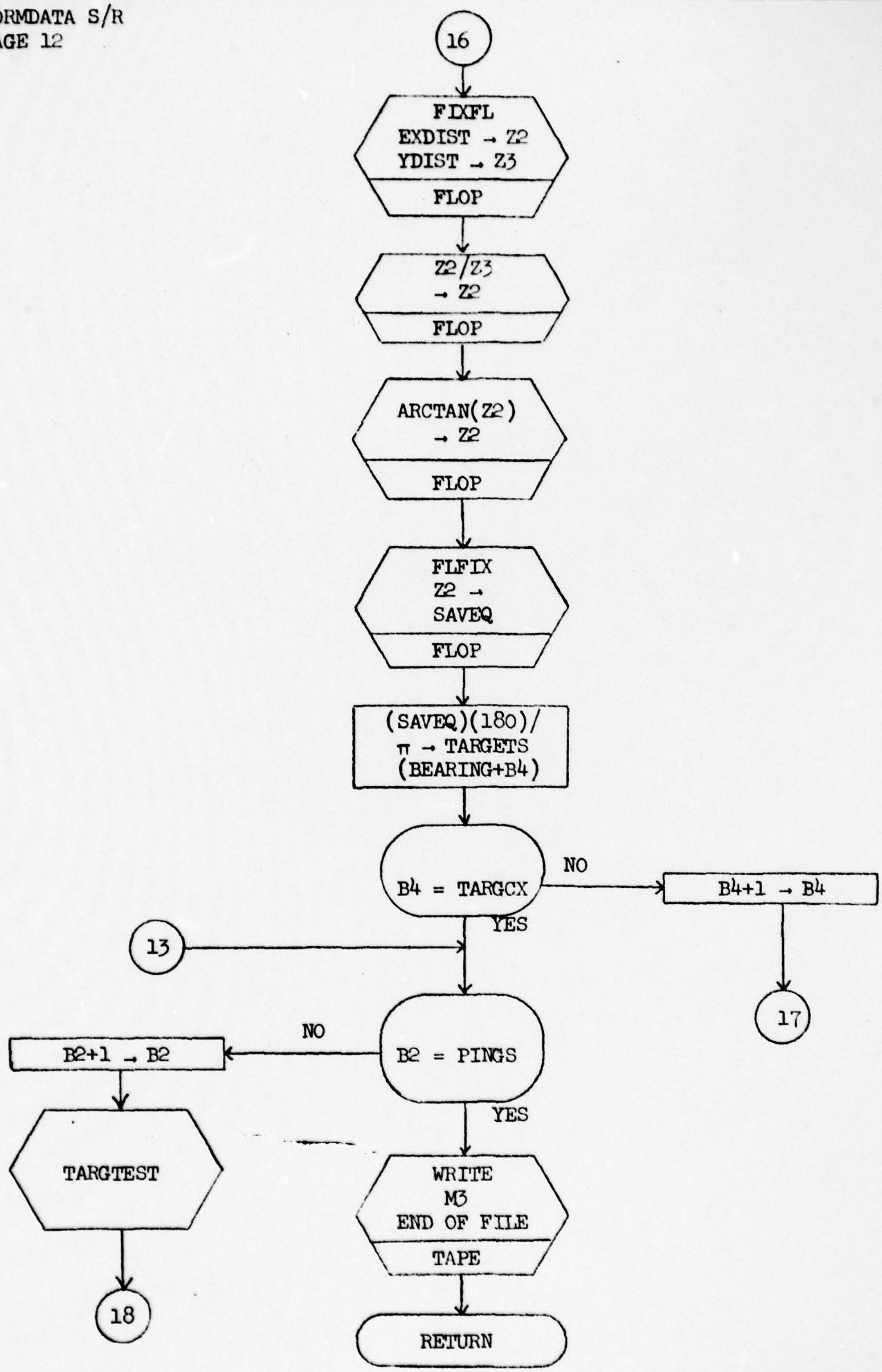
FORMDATA S/R
PAGE 8



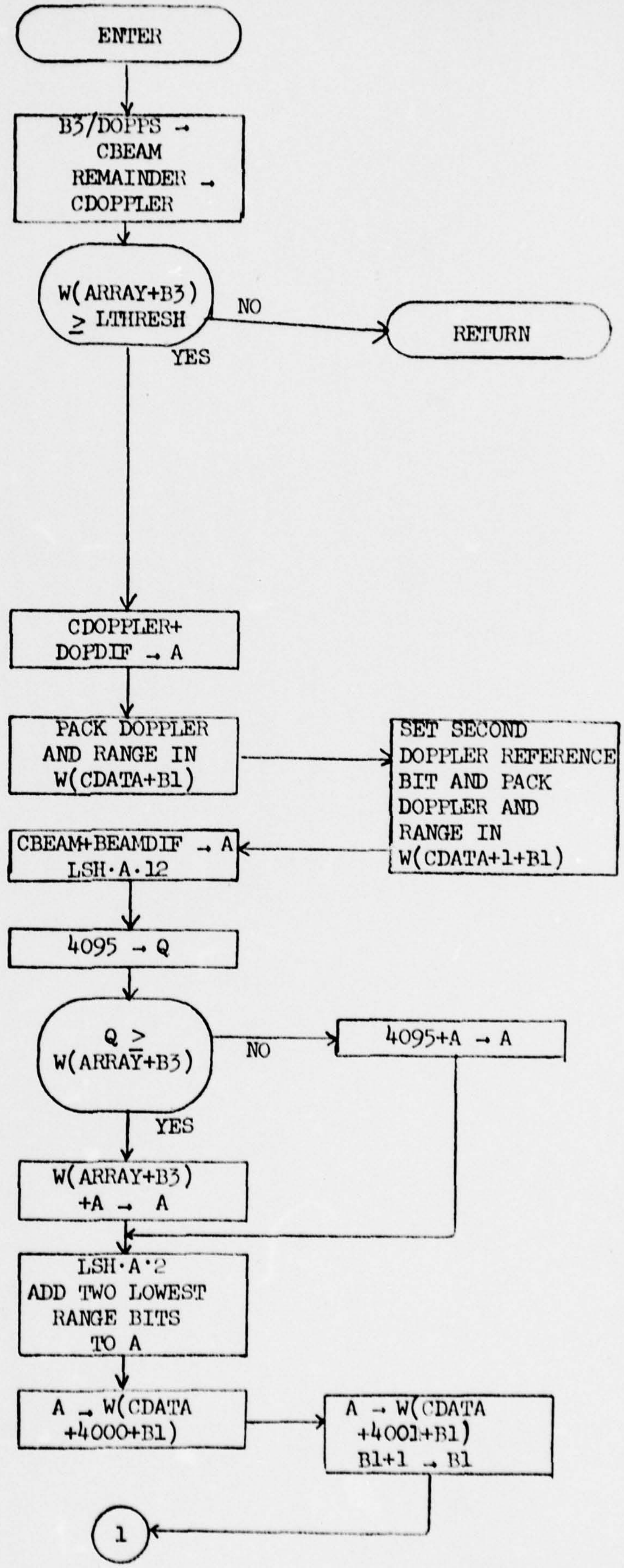

11
WRITE
M3
CDATA TABLE
TAPE
12
WRITE
M3
END OF FILE
TAPE
CLEAR CDATA
TABLE
0 → B1
TARGX = 0
YES
13
NO
0 → B4
17
TARGET(BEARING
+B4)·π/180
→ SAVEQ
FIXFL
SAVEQ
→ Z1
FLOP
Z1 →
SIN(Z2)
FLOP
Z1 →
SIN(Z3)
FLOP
14

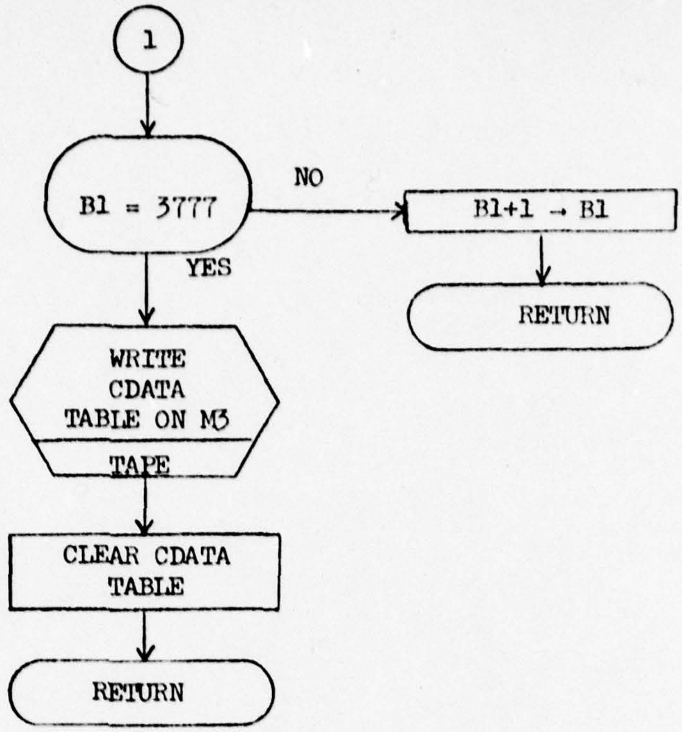

14
FLFIX
Z2 → SINPH1
Z3 → COSPH1
FLOP
(SINPH1) TARGETS
(RANGE+B4)
→ EXDIST
(COSPH1) TARGETS
(RANGE+B4)
→ YDIST
TARGETS(SPEED
+B4)(PTIME)
(0.194709) →
LDIST
TARGETS(COURSE
+B4)π/180 →
SAVEQ
FIXFL
SAVEQ
→ Z1
FLOP
Z1 →
SIN(Z2)
FLOP
Z1 →
COS(Z3)
FLOP
FLFIX
Z2 → SINPH1
Z3 → COSPH1
FLOP
15


15
(LDIST)(SINPHI)
+EXDIST →
EXDIST
(LDIST)(COSPHI)
+YDIST →
YDIST
FIXFL
EXDIST → Z2
YDIST → Z3
FLOP
(Z2)(Z2) →
Z2
FLOP
(Z3)(Z3)
→ Z3
FLOP
(Z2)(Z3)
→ Z2
FLOP
√Z2 → Z2
FLOP
FLFIX
Z2 → SAVEA
FLOP
SAVEA →
TARGET(RANGE+B4)
16

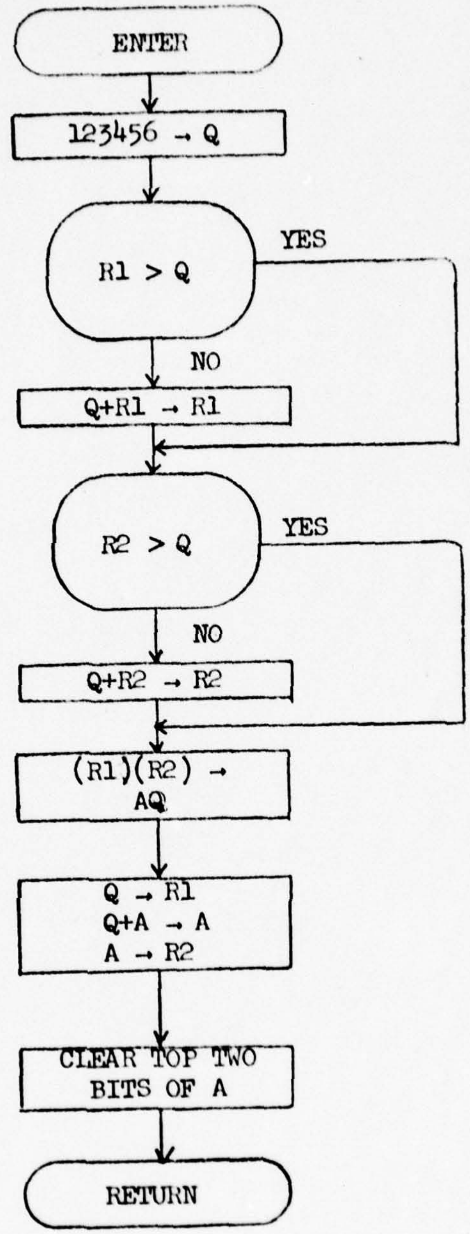

16
FIXFL
EXDIST → Z2
YDIST → Z3
FLOP
Z2/Z3
→ Z2
FLOP
ARCTAN(Z2)
→ Z2
FLOP
FLFIX
Z2 →
SAVEQ
FLOP
(SAVEQ)(180)/
π → TARGETS
(BEARING+B4)
B4 = TARGCX
NO
B4+1 → B4
17
YES
13
B2 = PINGS
NO
B2+1 → B2
TARGTEST
18
YES
WRITE
M3
END OF FILE
TAPE
RETURN

PACKITEM S/R

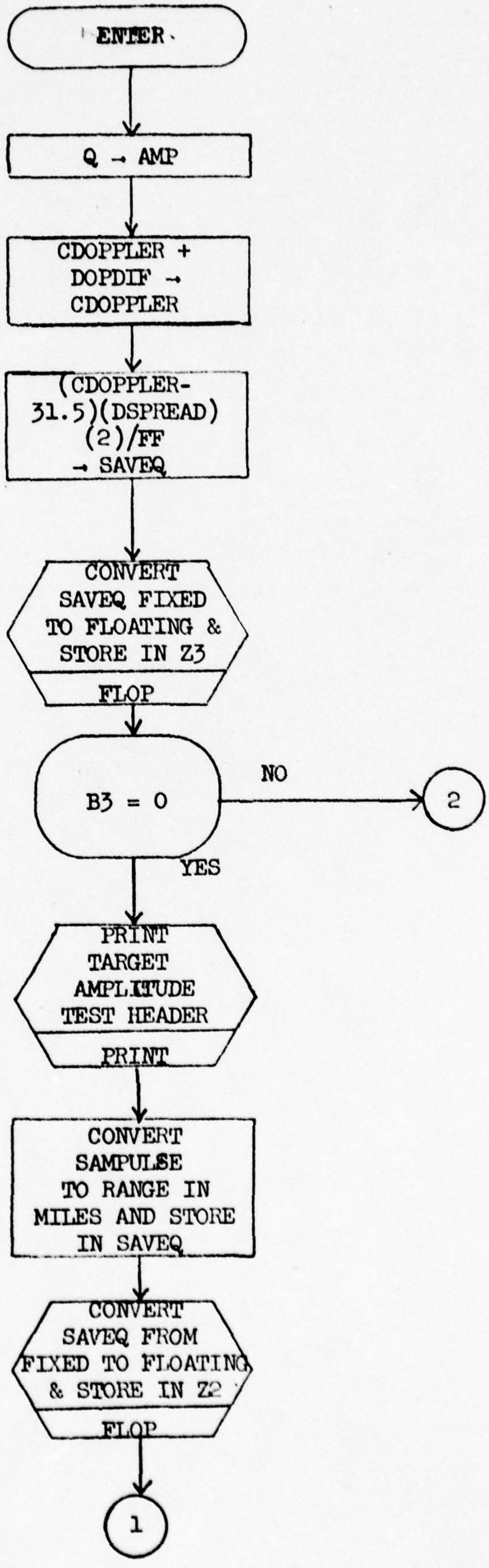

ENTER
B3/DOPPS → CBEAM
REMAINDER → CDOPPLER
W(ARRAY+B3) ≥ LTHRESH
NO
RETURN
YES
CDOPPLER+ DOPDIF → A
PACK DOPPLER AND RANGE IN W(CDATA+B1)
SET SECOND DOPPLER REFERENCE BIT AND PACK DOPPLER AND RANGE IN W(CDATA+1+B1)
CBEAM+BEAMDIF → A
LSH·A·12
4095 → Q
Q ≥ W(ARRAY+B3)
NO
4095+A → A
YES
W(ARRAY+B3) +A → A
LSH·A·2
ADD TWO LOWEST RANGE BITS TO A
A → W(CDATA +4000+B1)
A → W(CDATA +4001+B1)
B1+1 → B1
1


1
B1 = 3777
NO
B1+1 → B1
RETURN
YES
WRITE CDATA TABLE ON M3 TAPE
CLEAR CDATA TABLE
RETURN

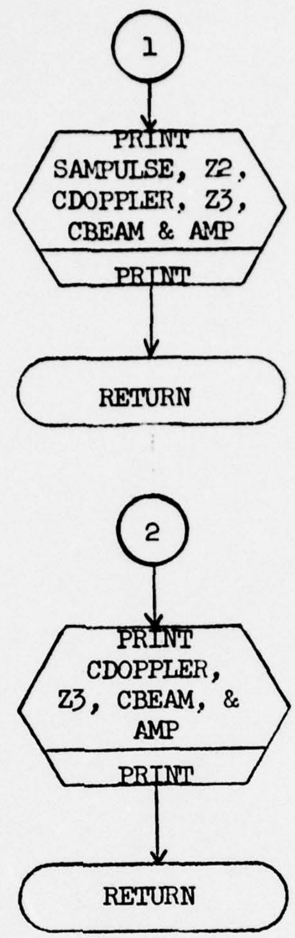
RANDOM S/R


ENTER
123456 → Q
R1 > Q
YES
NO
Q+R1 → R1
R2 > Q
YES
NO
Q+R2 → R2
(R1)(R2) → AQ
Q → R1
Q+A → A
A → R2
CLEAR TOP TWO BITS OF A
RETURN

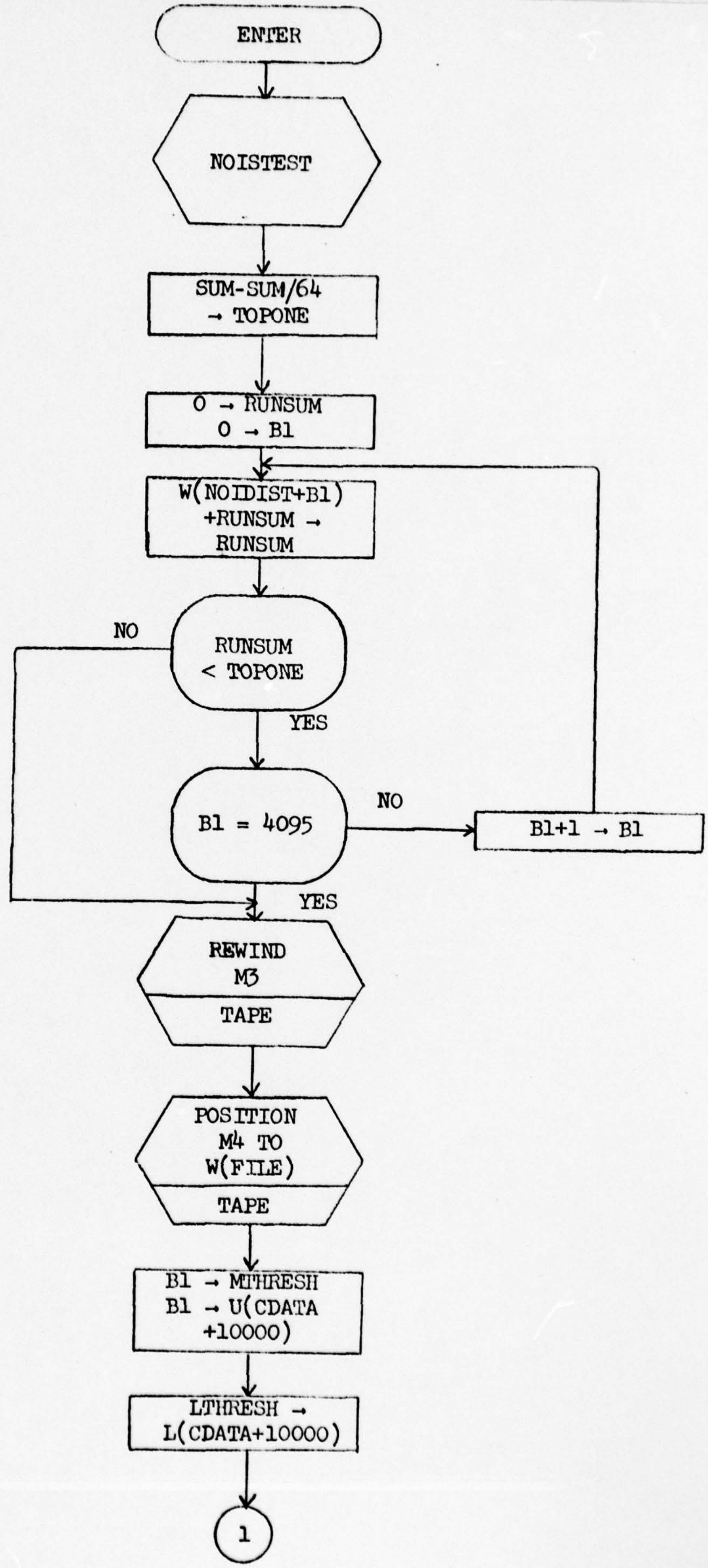
TARGAMPT S/R


ENTER
Q → AMP
CDOPPLER + DOPDIF → CDOPPLER
(CDOPPLER-31.5)(DSPREAD)(2)/FF → SAVEQ
CONVERT SAVEQ FIXED TO FLOATING & STORE IN Z3
FLOP
B3 = 0
NO
2
YES
PRINT TARGET AMPLITUDE TEST HEADER
PRINT
CONVERT SAMPULSE TO RANGE IN MILES AND STORE IN SAVEQ
CONVERT SAVEQ FROM FIXED TO FLOATING & STORE IN Z2
FLOP
1

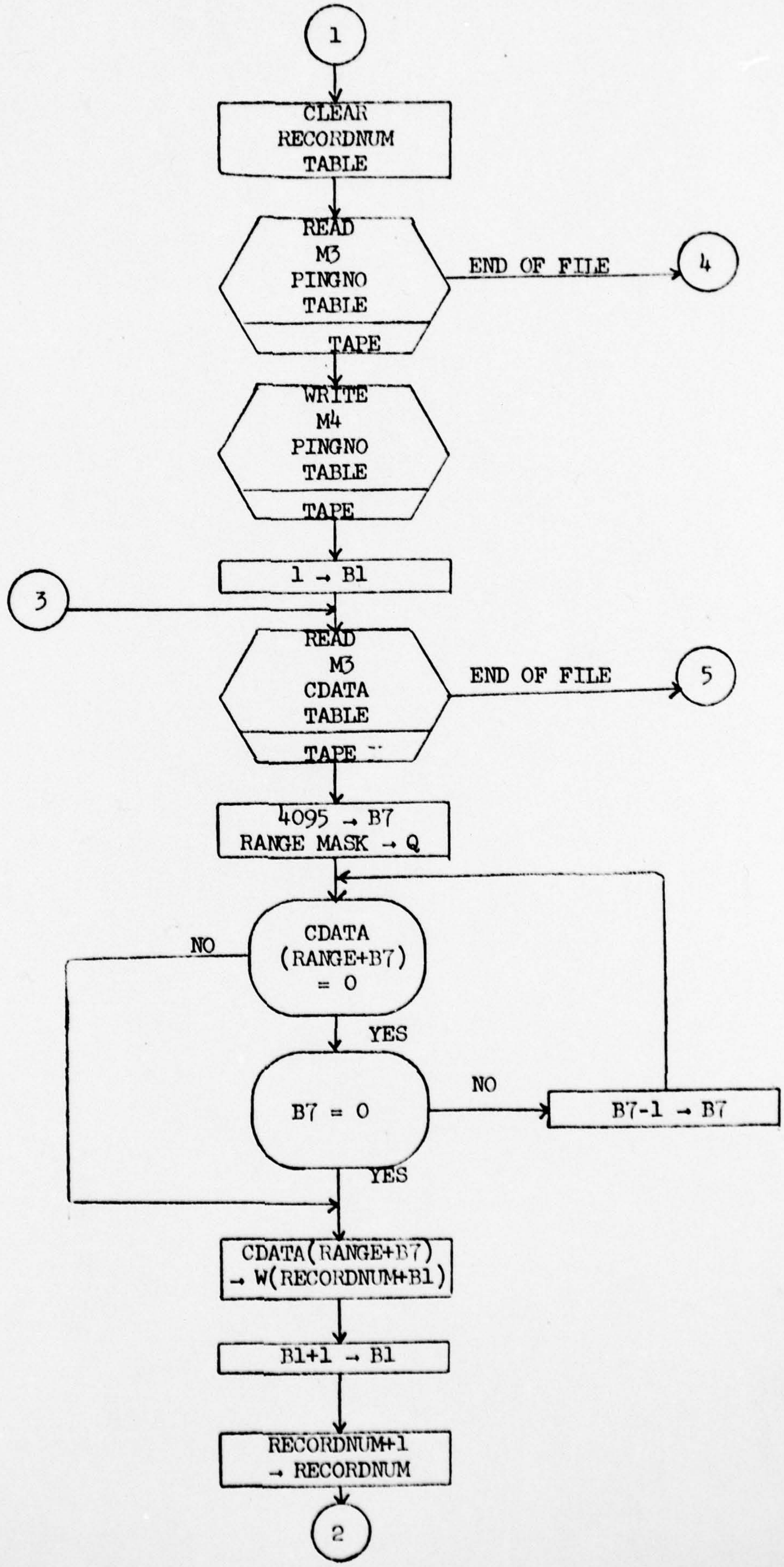
TARGAMPT
PAGE 2

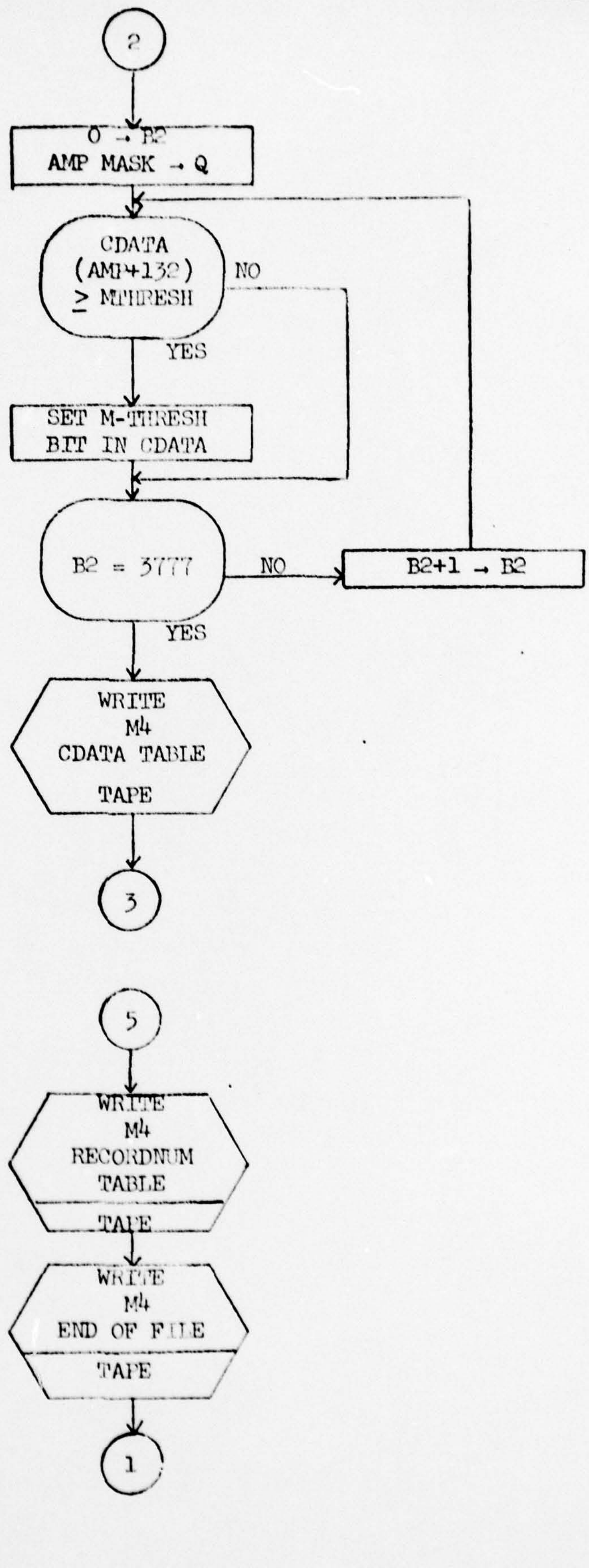


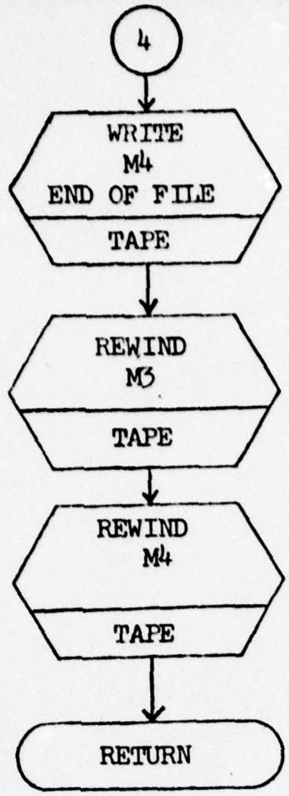
SETMTHRESH


ENTER
NOISTEST
SUM-SUM/64
→ TOPONE
0 → RUNSUM
0 → B1
W(NOIDIST+B1)
+RUNSUM →
RUNSUM
NO
RUNSUM
< TOPONE
YES
B1 = 4095
NO
B1+1 → B1
YES
REWIND
M3
TAPE
POSITION
M4 TO
W(FILE)
TAPE
B1 → MTHRESH
B1 → U(CDATA
+10000)
LTHRESH →
L(CDATA+10000)
1

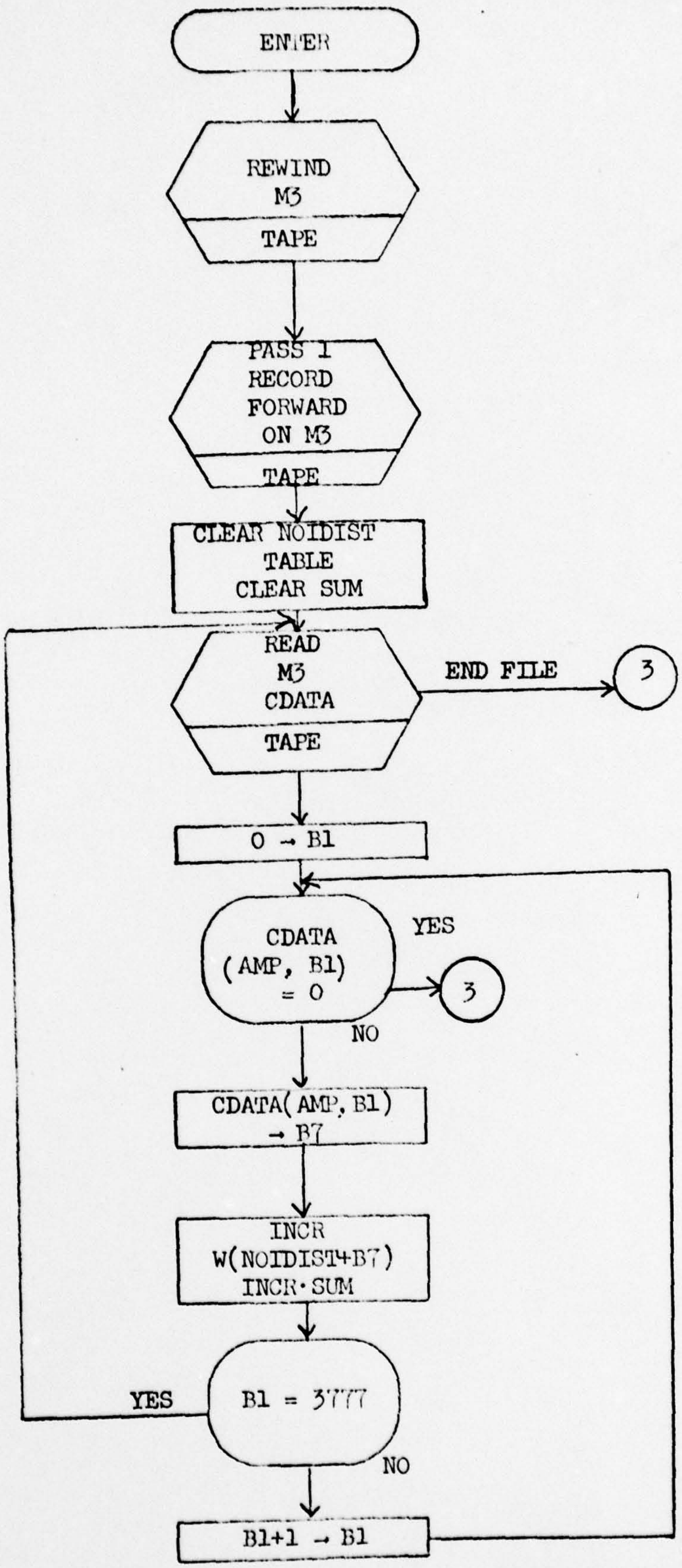
SETMTHRESH
PAGE 2



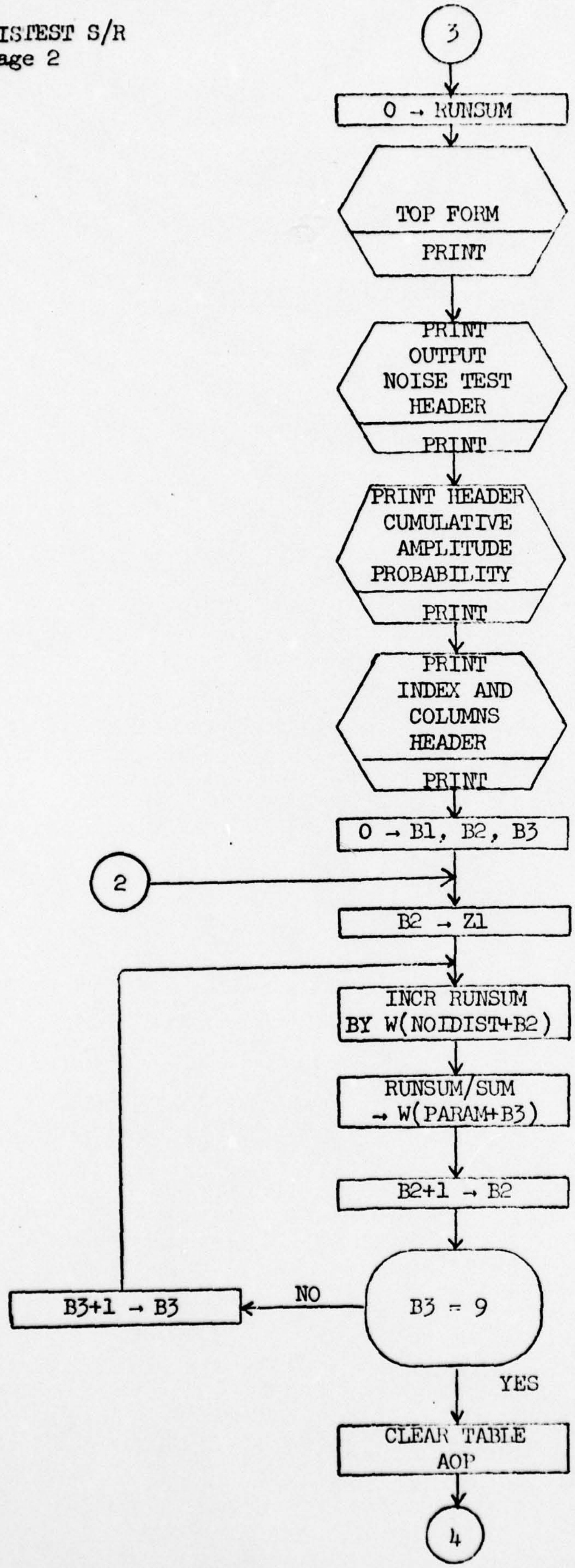

2
0 → B2
AMP MASK → Q
CDATA (AMP+132) ≥ MTHRESH
NO
YES
SET M-THRESH BIT IN CDATA
B2 = 3777
NO
B2+1 → B2
YES
WRITE M4 CDATA TABLE TAPE
3
5
WRITE M4 RECORDNUM TABLE TAPE
WRITE M4 END OF FILE TAPE
1


4
WRITE
M4
END OF FILE
TAPE
REWIND
M3
TAPE
REWIND
M4
TAPE
RETURN

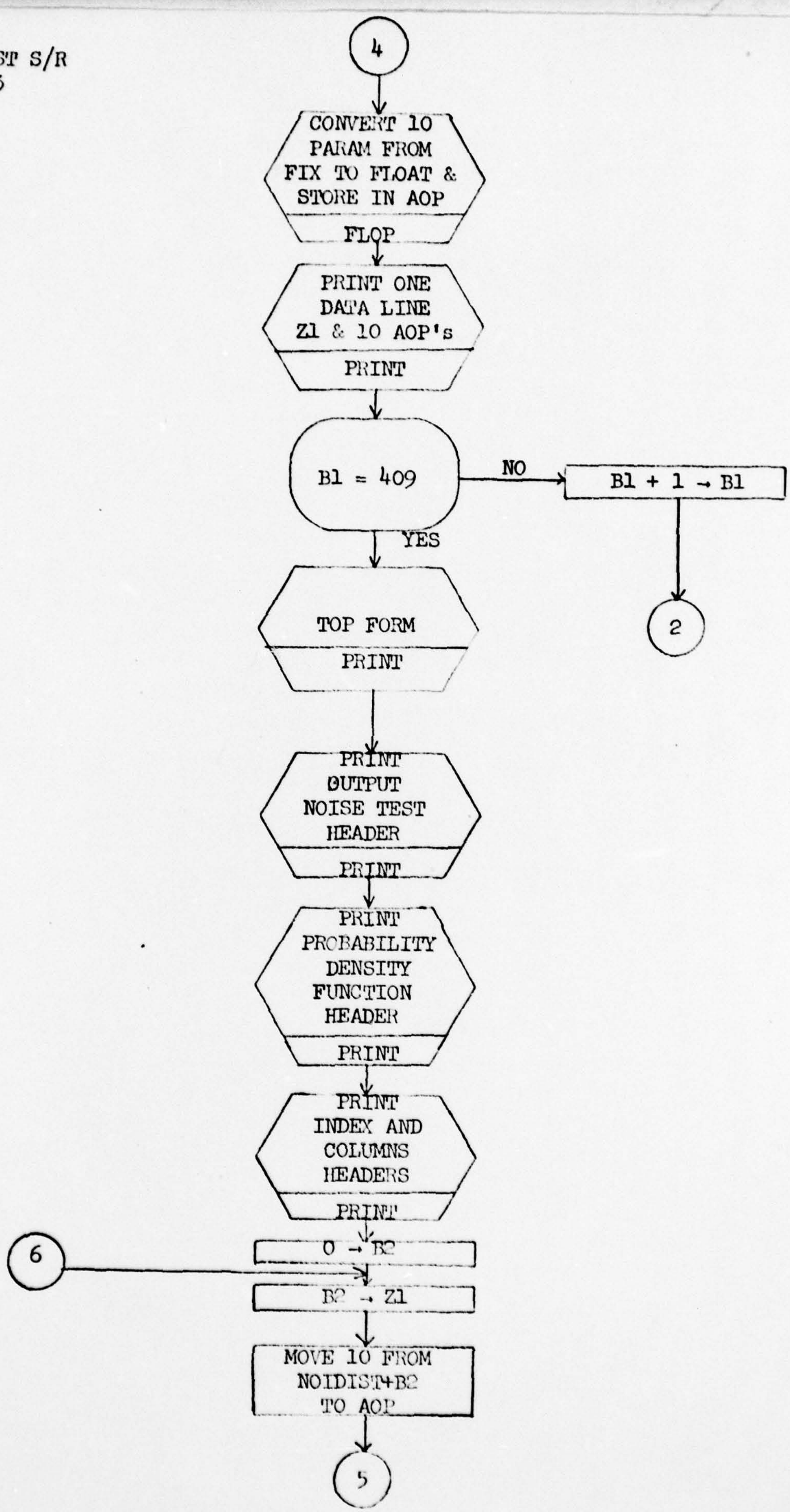
NOISTEST S/R

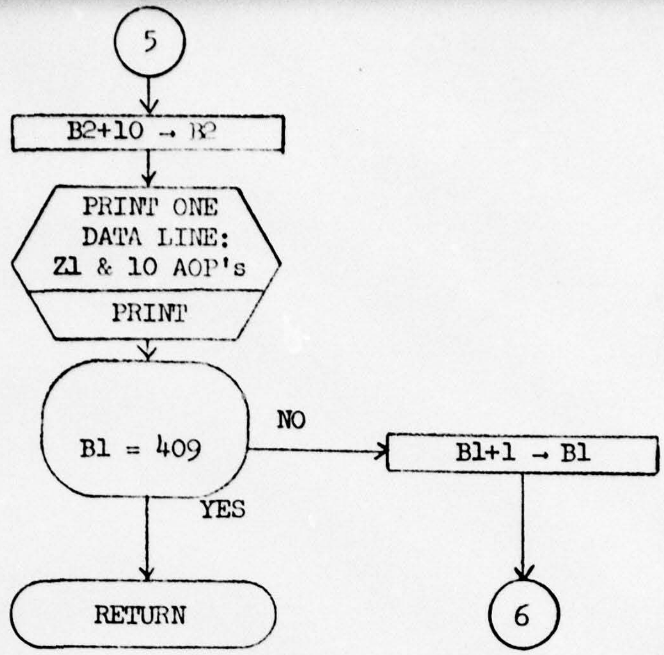

ENTER
REWIND
M3
TAPE
PASS 1
RECORD
FORWARD
ON M3
TAPE
CLEAR NOIDIST
TABLE
CLEAR SUM
READ
M3
CDATA
TAPE
END FILE
3
0 → B1
CDATA
(AMP, B1)
= 0
YES
NO
3
CDATA(AMP, B1)
→ B7
INCR
W(NOIDIST+B7)
INCR·SUM
YES
B1 = 3777
NO
B1+1 → B1

NOISTEST S/R
page 2

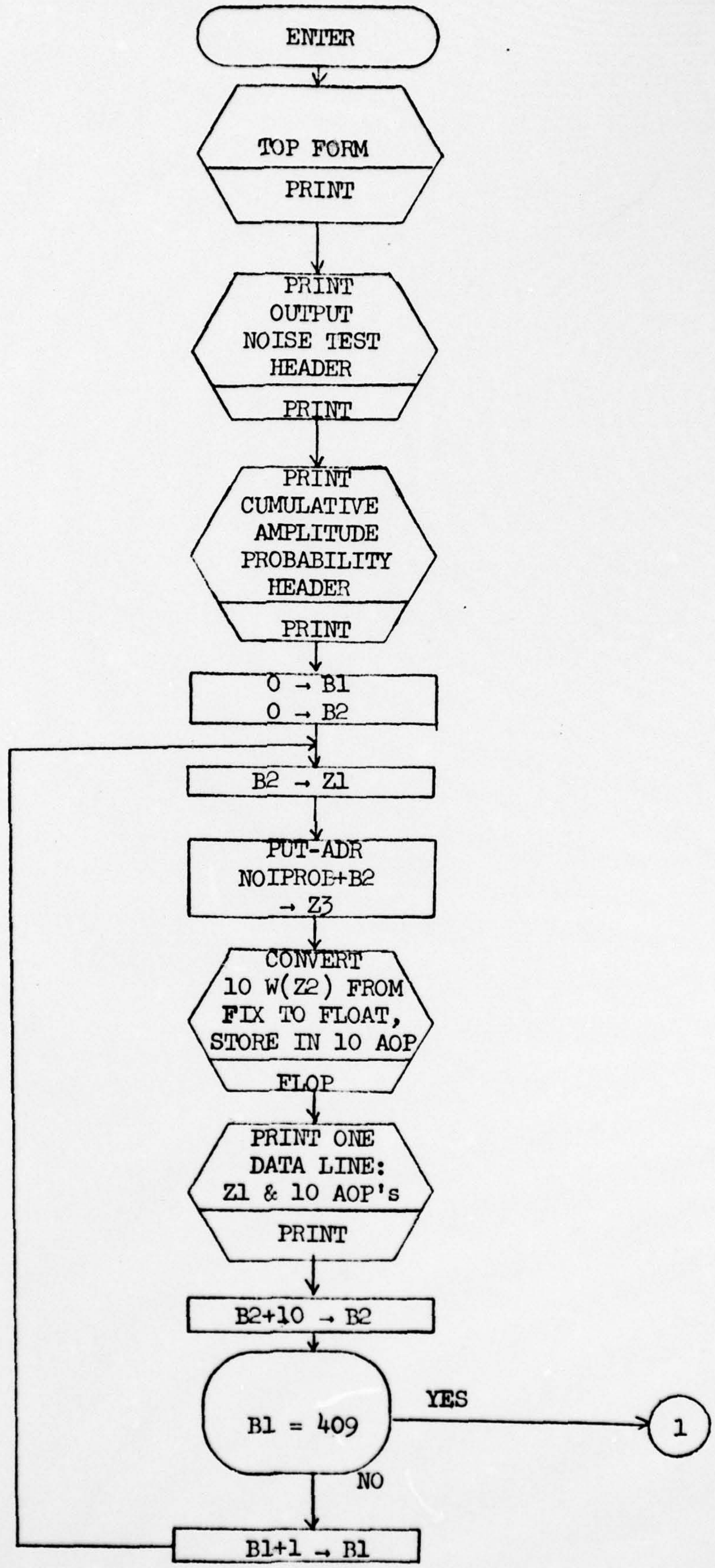


NOISTEST S/R
PAGE 3

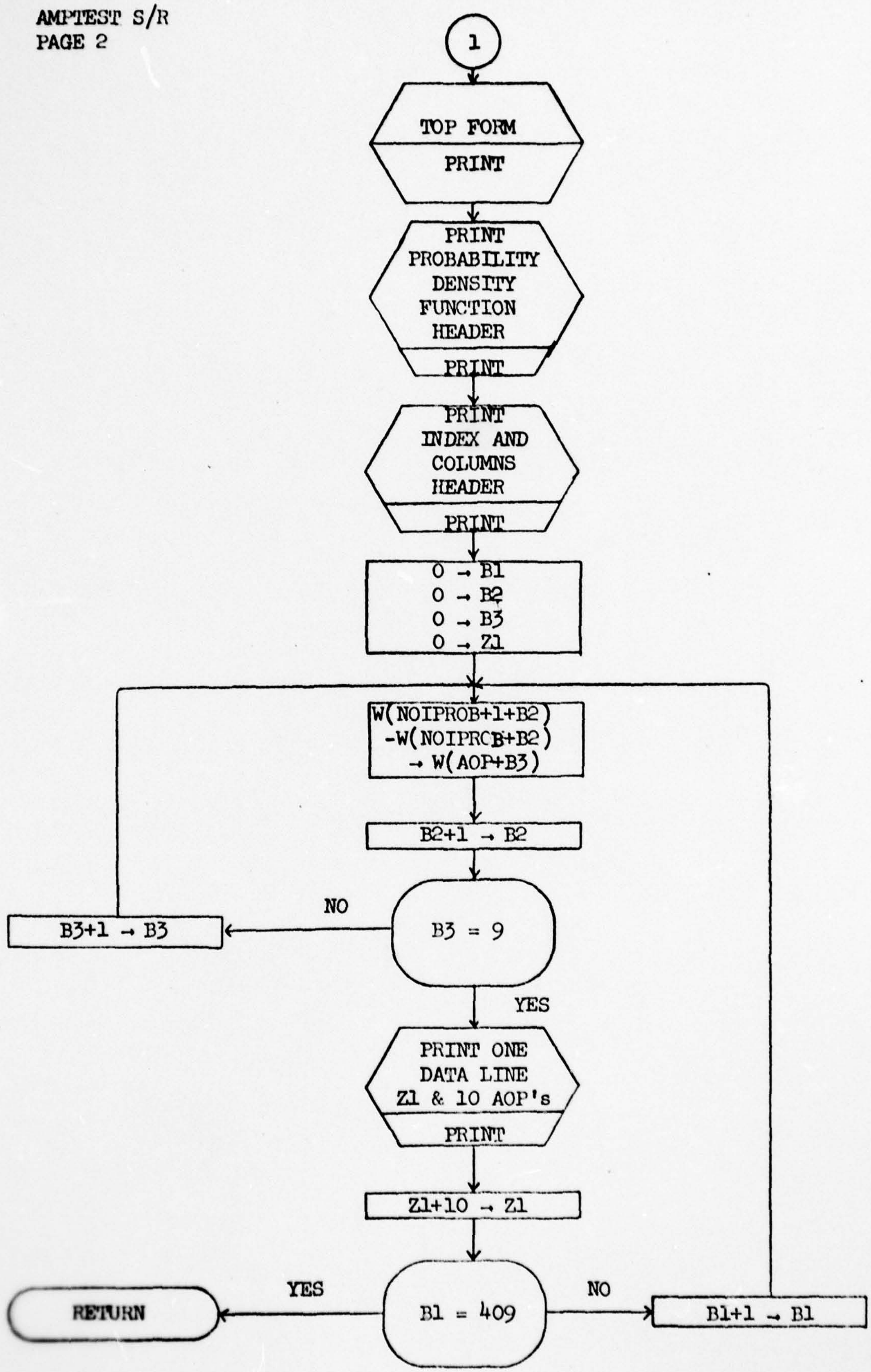



5
B2+10 → B2
PRINT ONE
DATA LINE:
Z1 & 10 AOP's
PRINT
B1 = 409
NO
B1+1 → B1
YES
RETURN
6

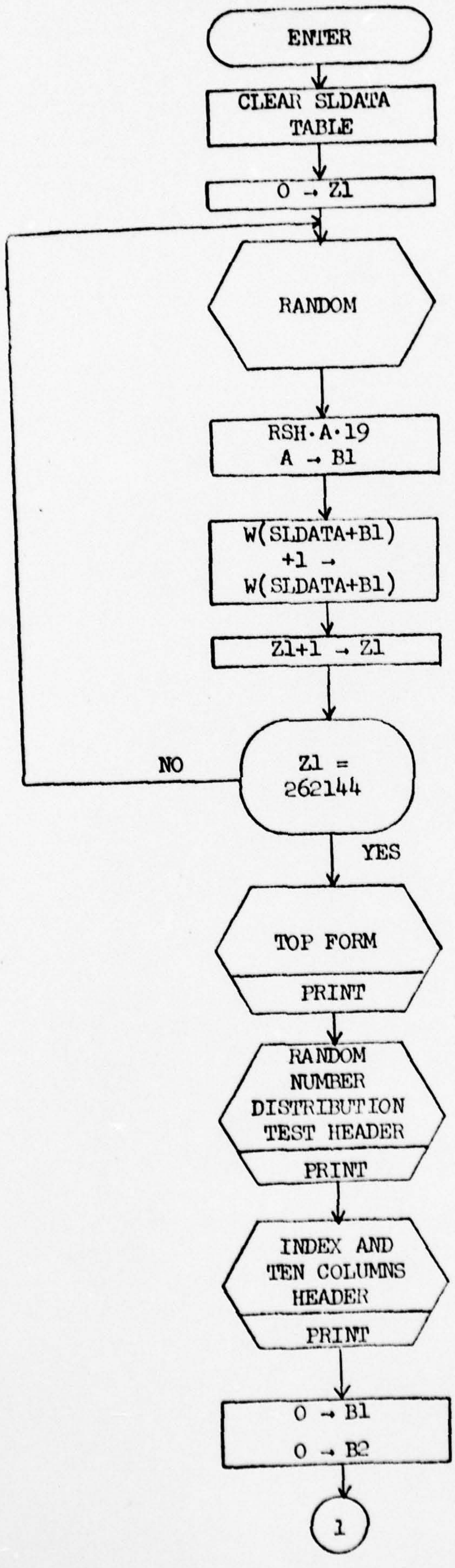
AMPTEST S/R

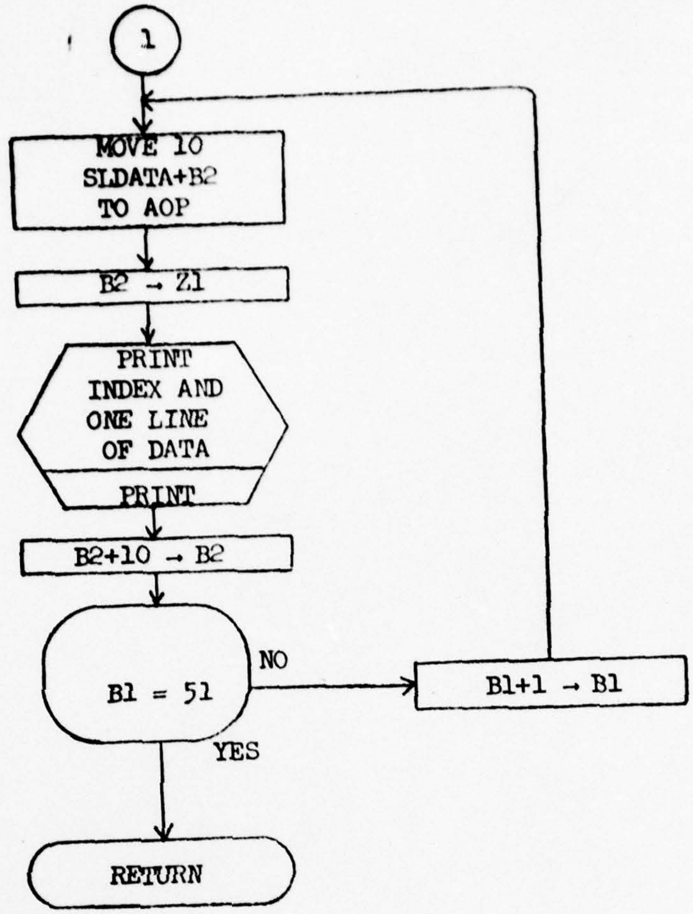

ENTER
TOP FORM
PRINT
PRINT OUTPUT NOISE TEST HEADER
PRINT
PRINT CUMULATIVE AMPLITUDE PROBABILITY HEADER
PRINT
0 → B1
0 → B2
B2 → Z1
PUT-ADR NOIPROB+B2 → Z3
CONVERT 10 W(Z2) FROM FIX TO FLOAT, STORE IN 10 AOP
FLOP
PRINT ONE DATA LINE: Z1 & 10 AOP's
PRINT
B2+10 → B2
B1 = 409
YES
1
NO
B1+1 → B1

AMPTEST S/R
PAGE 2

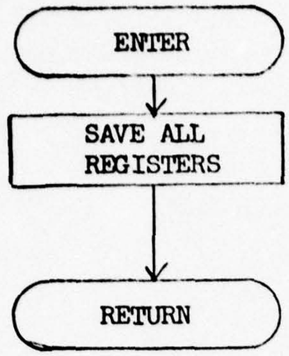


RANDOMTEST S/R

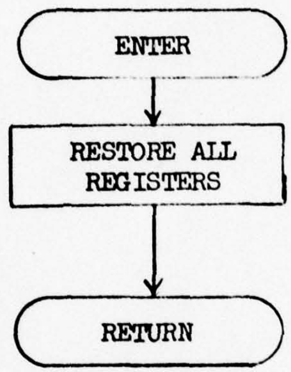



1
MOVE 10
SLDATA+B2
TO AOP
B2 → Z1
PRINT
INDEX AND
ONE LINE
OF DATA
PRINT
B2+10 → B2
B1 = 51
NO
B1+1 → B1
YES
RETURN

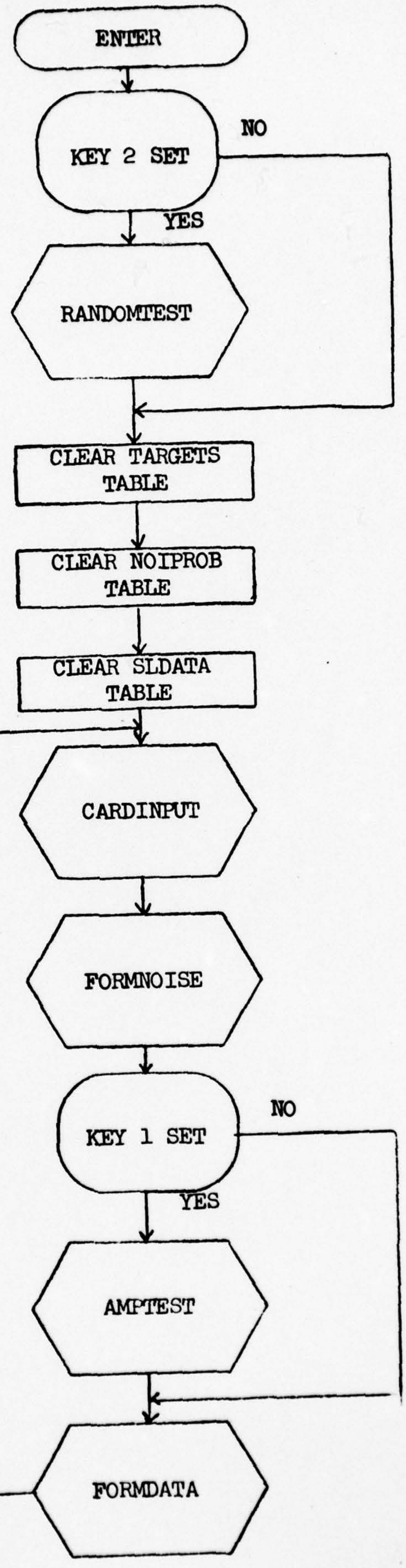
SAVEREG S/R

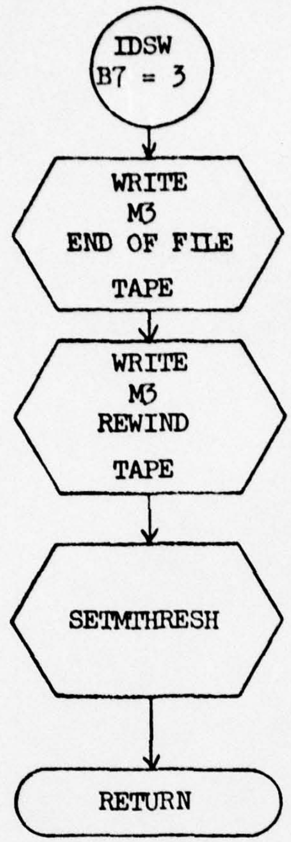


RESTREG S/R



SIMULATASP


ENTER
KEY 2 SET
NO
YES
RANDOMTEST
CLEAR TARGETS TABLE
CLEAR NOIPROB TABLE
CLEAR SLDATA TABLE
CARDINPUT
FORMNOISE
KEY 1 SET
NO
YES
AMPTEST
FORMDATA


IDSW
B7 = 3
WRITE
M3
END OF FILE
TAPE
WRITE
M3
REWIND
TAPE
SETMTHRESH
RETURN